UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
PROGRAMA DE PÓS GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Marcos Aurélio Pedroso Leandro

FEDERAÇÃO DE IDENTIDADES E COMPUTAÇÃO EM NUVEM: ESTUDO DE CASO USANDO SHIBBOLETH

Dissertação submetida ao Programa de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Ciência da Computação.
Orientadora: Profª. Dra. Carla Merkle Westphall

Florianópolis

2012

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática da Biblioteca Universitária da UFSC.

Leandro, Marcos Aurélio Pedroso
Federação de Identidades e Computação em Nuvem [dissertação] : Estudo de Caso Usando Shibboleth / Marcos Aurélio Pedroso Leandro ; orientadora, Carla Merkle Westphall - Florianópolis, SC, 2012.
120 p. ; 21cm

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Ciência da Computação. 2. Computação em Nuvem. 3. Gerenciamento de Identidades. 4. Federação. 5. Shibboleth. I. Westphall, Carla Merkle. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Marcos Aurélio Pedroso Leandro

FEDERAÇÃO DE IDENTIDADES E COMPUTAÇÃO EM NUVEM: ESTUDO DE CASO USANDO SHIBBOLETH

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Ciência da Computação na área de concentração Sistemas de Computação, e aprovada em sua forma final pelo Programa de Pós Graduação em Ciência da Computação

Florianópolis, 30 de março de 2012.

_____
Prof. Dr. Ronaldo dos Santos Mello
Coordenador do Curso

Banca Examinadora:

_____
Prof.ª Dr.ª Carla Merkle Westphall
Orientadora
Universidade Federal de Santa Catarina

_____
Prof. Dr. Altair Olivo Santin
Pontifícia Universidade Católica do Paraná

_____
Prof. Dr. João Bosco Mangueira Sobral
Universidade Federal de Santa Catarina

_____
Prof.ª Dr.ª Patrícia Della Méa Plentz
Universidade Federal de Santa Catarina

Aos amigos.

AGRADECIMENTOS

Agradeço a Deus pela vida, força, vontade e disciplina e por sua guia e proteção.

Agradeço aos pais e irmãos pela força e motivação para dar continuidade nesta caminhada.

Agradeço a minha orientadora, professora Carla, e seu esposo, professor Carlos Becker Westphall, pela orientação e pelo incentivo.

Agradeço aos amigos, em especial à Lilian Berton, à Lenira Fernandes Ferreira e ao Claudemir Siburski, pela grande amizade e apoio que recebi em todos os sentidos.

Agradeço à banca composta pelos professores João Bosco, Altair Santin e Patrícia Plentz pelo tempo que disponibilizaram para a participação na avaliação do trabalho.

Agradeço aos órgãos de fomento à pesquisa e a todos os colaboradores do programa de pós-graduação em Ciência da Computação da UFSC, bem como à PRPG, por estarem juntos tornando possível o progresso na caminhada acadêmica.

Quae sunt Caesaris, Caesari.

RESUMO

Os serviços disponibilizados em nuvens podem representar um aumento na eficiência e eficácia na operação dos negócios empresariais, melhorando o custo-benefício em relação ao consumo de recursos e serviços. Os sistemas de computação em nuvem possuem muitas vantagens se comparados aos serviços prestados tradicionalmente, como investimento inicial reduzido, alta disponibilidade, escalabilidade infinita, grande capacidade de tolerância a falhas, entre outros benefícios. Entretanto, existe uma preocupação em relação à privacidade dos dados, uma vez que esses dados encontram-se fora do domínio do cliente. Para que esses serviços sejam efetivamente usados pelas organizações é necessário prover controle de acesso. Questões de segurança devem ser consideradas para prover a autenticidade, confidencialidade e integridade do ambiente como um todo. A implantação de um modelo de gerenciamento seguro e confiável se faz necessária. Neste sentido, o gerenciamento de identidades, bem como os mecanismos de federação de identidades, são fundamentais para alcançar os objetivos de segurança na nuvem. Um sistema de gerenciamento de identidades é composto de protocolos e componentes de software que tratam as identidades dos indivíduos durante todo o ciclo de vida de suas identidades. Estes sistemas possibilitam a criação de federações. Uma federação é uma forma de associação de parceiros de uma rede colaborativa que possibilita a cooperação entre os membros da federação. O gerenciamento de identidades federadas permite que as organizações de uma federação interajam com base na gestão da identidade compartilhada, permitindo, dessa forma, a autenticação única (*Single Sign-On, SSO*). O objetivo deste trabalho é apresentar um estudo de caso do uso da ferramenta Shibboleth, destacando o conceito de gerenciamento de identidades federadas em um ambiente de nuvem computacional. Esta ferramenta dá apoio às tarefas de gerenciamento de identidades e permite a criação de federações. Entretanto, neste trabalho será abordado seu uso especificamente em um ambiente de nuvem. Como resultado foi obtido a implementação de uma parte do cenário proposto e posteriormente testado em dois casos de uso, o primeiro para acesso público e o segundo para acesso privado. Os testes tiveram os resultados esperados.

Palavras-chave: Computação em Nuvem; Gerenciamento de Identidades; Federação; *Single Sign-on*; SSO; Shibboleth; Controle de Acesso; Autenticação; Autorização.

ABSTRACT

The services provided in clouds may represent an increase in efficiency and effectiveness in the operation of the enterprise business, improving the cost-effective related to services and resources consumption. The cloud computing systems have many advantages compared to traditional services, such as reduced upfront investment, expected performance, high availability, infinite scalability, tremendous fault-tolerance capability, and other benefits. However, there is concern about privacy of data, since these data are outside of the client domain. For these services to be effectively enjoyed by organizations is necessary to provide access control. Security issues should be considered to provide authenticity, confidentiality and integrity of the environment as a whole. The implementation of a management model safe and reliable is needed. In this sense, identity management, as well as mechanisms for identity federation, are critical to achieving the objectives of security in the cloud. An identity management system is composed of protocols and software components that handle the identities of individuals throughout the life cycle of their identities. These systems allow the creation of federations. A federation is a form of association of partners in a collaborative network that allows the cooperation between members of the federation. The federated identity management allows organizations in a federation to interact based on the management of shared identity, allowing thus the Single Sign-On. The purpose of this work is to present a case study using the tool Shibboleth highlighting the concept of federated identity management in a cloud computing environment. This tool supports the tasks of identity management and allows the creation of federations. However, this work will be specifically addressed its use in a cloud environment. The result was the implementation of a part of the scenario proposed and then tested in two cases of use, the first for public access and the second for private access. The tests had the expected results.

Keywords: Cloud Computing; Identity Management; Multi-Tenancy; Federation; Shibboleth; Access Control; Authentication; Authorization.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AA - *Attribute Authority*
ABAC - *Attribute-Based Access Control*
ACS - *Assertion Consumer Service*
Amazon EBS - *Amazon Elastic Block Store*
Amazon S3 - *Amazon Simple Storage Service*
AMI - *Amazon Machine Image*
AR - *Attribute Requester*
AWS - *Amazon Web Services*
CD SSO - *Cross Domain Single Sign-On*
CRM - *Customer Relationship Management*
CSA - *Cloud Security Alliance*
CSP - *Cloud Service Provider*
DaaS - *Data as a Service*
EC2 - *Elastic Compute Cloud*
FIM - *Federated Identity Management*
GIA - *Gerenciamento de Identidade e Acesso*
HRM - *Human Resources Management*
HS - *Handle Service*
HTTP - *Hypertext Transfer Protocol*
HTTPS - *Hypertext Transfer Protocol Secure*
IaaS - *Infrastructure as a Service*
IAM - *Identity and Access Management*
IDaaS - *Identity as a Service*
IDM - *Identity Management*
IdP- *Identity Provider*
IMS - *Identity Management System*
IPMaaS - *Identity and Policy Management as a Service*
ITU - *International Telecommunication Union*
JADE - *Java Agent DEvelopment Framework*
JDBC - *Java Database Connectivity*
JDK - *Java Development Kit*
JEE - *Java Enterprise Edition*
LDAP - *Lightweight Directory Access Protocol*
MACE - *Middleware Architecture Committee for Education*
MPCC - *Mutual Protection for Cloud Computing*
MRC - Monitor de Referência do Consumidor
MRP - Monitor de Referência do Provedor

NaaS - *Network as a Service*
NIST - *National Institute of Standards and Technology*
OASIS - *Organization for the Advancement of Structured Information Standards*
PaaS - *Platform as a Service*
PAOS – *Reverse SOAP*
PDA - *Personal Digital Assistant*
PDP - *Policy Decision Point*
PII - *Personally Identifiable Information*
QoS - *Quality of Service*
RBAC - *Role-Based Access Control*
RDP - *Remote Desktop Protocol*
REST - *Representational State Transfer*
RM - *Resource Manager*
RPC - Repositório de Política do Consumidor
RPP - Repositório de Política do Provedor
SaaS - *Software as a Service*
SAML - *Security Assertion Markup Language*
SLA - *Service Level Agreement*
SOA - *Service Oriented Architecture*
SOAP - *Simple Object Access Protocol*
SP - *Service Provider*
SPI - *Software, Platform and Infrastructure (as a Service)*
SPML - *Service Provisioning Markup Language*
SQL - *Structured Query Language*
SSH - *Secure Shell*
SSL - *Secure Sockets Layer*
SSO - *Single Sign-On*
SSTC - *Security Services Technical Committee*
STS - *Security Token Service*
TTP - *Trusted Third Party*
URI - *Uniform Resource Identifier*
URL - *Uniform Resource Locator*
UUID - *Universally Unique Identifier*
VME - *Virtual Machine Environment*
WAYF - *Where Are You From*
XACML - *eXtensible Access Control Markup Language*
XML - *eXtensible Markup Language*
XSS - *Cross-site Scripting*

SUMÁRIO

1 INTRODUÇÃO

O conceito de computação em nuvem permite a utilização de serviços e recursos sob demanda e vem ganhando atenção tanto de empresas como de usuários individuais. Neste modelo podem ser oferecidos serviços relacionados às capacidades computacionais, como por exemplo, máquinas virtuais, armazenamento e sistemas operacionais completos. Estes serviços são divididos em: infraestrutura (*Infrastructure as a Service*, IaaS); plataforma (*Platform as a Service*, PaaS) e aplicações (*Software as a Service*, SaaS). Cada uma dessas camadas pode possuir a capacidade de oferecer serviços específicos, como é o caso de IPMaaS (*Identity and Policy Management as a Service*), NaaS (*Network as a Service*) e DaaS (*Data as a Service*), entre muitos outros (ZHOU, ZHANG, *et al.*, 2010).

Os serviços disponibilizados em nuvens podem representar um aumento na eficiência e eficácia na operação dos negócios empresariais, melhorando o custo-benefício em relação ao consumo de recursos e serviços. Os sistemas de computação em nuvem possuem muitas vantagens se comparados aos serviços prestados tradicionalmente, como investimento inicial reduzido, alta disponibilidade, escalabilidade infinita, grande capacidade de tolerância a falhas, entre outros benefícios (ZHOU, ZHANG, *et al.*, 2010). Empresas como Salesfoce.com e Google desenvolvem e oferecem serviços em nuvem, enquanto muitas companhias e entidades governamentais consideram a construção de centros de dados como nuvem privada ou a integração de serviços de nuvem dentro de suas estruturas (JUNIPER NETWOKS, 2009).

Entretanto, existe uma preocupação em relação à privacidade dos dados, uma vez que esses dados encontram-se fora do domínio do cliente. Ou seja, de um lado temos as vantagens dos serviços disponibilizados e, de outro, temos a preocupação com a segurança. Para que esses serviços sejam efetivamente usados pelas organizações é necessário prover controle de acesso. Assim, se faz necessário criar formas para impedir o acesso não autorizado a informações e que os dados sensíveis permaneçam privados. Questões de segurança devem ser consideradas para prover a autenticidade, confidencialidade e integridade. No que diz respeito à confiabilidade e responsabilidade, o provedor deve fornecer recursos confiáveis, especialmente se a computação a ser realizada for crítica e existindo uma clara delimitação de responsabilidade (SOUSA, MOREIRA e MACHADO, 2009).

Para que as organizações consumidoras utilizem os serviços oferecidos pela nuvem é necessária a implantação de um modelo de gerenciamento seguro e confiável. O esquema a ser utilizado deve facilitar a inserção e remoção de usuários dos serviços oferecidos pela nuvem. A implantação de mecanismos de autenticação robustos e esquemas de delegação de direitos funcionando de maneira confiável são fundamentais para o correto gerenciamento de identidades e para a prestação de serviços em nuvens computacionais.

O gerenciamento de identidades é definido como sendo um conjunto de funções e capacidades, como administração, gerência e manutenção, descoberta, troca de informação, aplicação de políticas e autenticação, usadas para garantir as informações da identidade, garantindo dessa maneira a segurança (CHADWICK, 2009). A gestão de identidade é utilizada para controlar o acesso a qualquer sistema/recurso através dos direitos associados ao usuário e as restrições estabelecidas para a identidade (BELAPURKAR, CHAKRABARTI, *et al.*, 2009).

Os sistemas de gerenciamento de identidades são importantes no atual contexto empresarial para gerir um grande número de usuários. Uma das principais vantagens desses sistemas é que, quando novas aplicações são fornecidas em uma empresa, eles podem aproveitar os dados do sistema de gerenciamento de identidades, sem a necessidade de dados específicos para a nova aplicação (BELAPURKAR, CHAKRABARTI, *et al.*, 2009).

A predominância de alianças nos negócios necessita maior evolução do gerenciamento de identidades, chamado de gerenciamento de identidades federadas (*Federated Identity Management*, FIM). A principal motivação do FIM é facilitar a federação de identidades entre os parceiros de negócios com ênfase na facilidade de gerenciamento de usuários (AHN e LAM, 2005). Federação de identidades é a chave para alcançar harmonia entre competência de negócio e eficiência tecnológica, e deve ser considerado o primeiro passo em direção à adoção de computação em nuvem (JUNIPER NETWOKS, 2009).

Uma federação é uma forma de associação de parceiros de uma rede colaborativa que usa um conjunto comum de atributos, práticas e políticas para trocar informações e compartilhar serviços, possibilitando a cooperação entre os membros da federação (CARMODY, ERDOS, *et al.*, 2005). O gerenciamento de identidades federadas, baseado nestes acordos comerciais, técnicos e políticos, permite que as organizações de uma federação interajam com base na gestão da identidade compartilhada (BUECKER, FILIP, *et al.*, 2005). Uma das funções

básicas oferecidas por essas soluções de federação de identidades é a autenticação única (*Single Sign-On* - SSO) (MALER e REED, 2008). Esta autenticação traz facilidades para os usuários, pois permite que esses passem pelo processo de autenticação uma única vez e usufruam das credenciais obtidas por todos os serviços que desejarem acessar (MELLO, WANGHAM, *et al.*, 2009).

O objetivo final da federação de identidades é permitir aos usuários de um domínio o acesso seguro de dados ou sistemas de outros domínios diretamente, sem a necessidade de administração redundante de usuários. A ferramenta Shibboleth (SCAVO e CANTOR, 2005)é um exemplo de solução que provê o gerenciamento de identidades federadas.

1.1 OBJETIVOS

1.1.1 Objetivo geral

O objetivo geral deste trabalho é apresentar um estudo de caso do uso da ferramenta Shibboleth, destacando o conceito de gerenciamento de identidades federadas em um ambiente de nuvem computacional.

1.1.2 Objetivos específicos

Os objetivos específicos deste trabalho são os seguintes:

- realizar um estudo para verificar o estado da arte sobre gerenciamento de identidades, computação em nuvens e modelos de segurança para nuvens baseados em federação;
- apresentar uma visão geral sobre os esforços para solucionar os problemas de segurança na nuvem;
- definir um cenário em nuvem para o uso da ferramenta Shibboleth;
- definir os casos de uso do cenário proposto;
- realizar testes com a ferramenta já em funcionamento dentro do cenário;
- mostrar resultados em relação aos testes realizados no cenário.

1.2 MOTIVAÇÃO

O gerenciamento de identidades e controle de acesso para aplicações corporativas permanece um dos maiores desafios enfrentados pela TI atualmente. Mesmo que uma empresa possa viabilizar vários serviços de computação em nuvem sem uma boa estratégia de gerenciamento de identidade e acesso, em longo prazo, estender os serviços de identidade da empresa à nuvem será um requisito necessário para o uso de serviços de computação sob demanda (CSA, 2010b).

Existem esforços na área, mas carecem de reforço e maturidade para evoluírem. Além do mais, computação em nuvem trata-se de um ambiente completamente heterogêneo, o que abre o leque de possibilidades para se tratar de segurança em diferentes níveis. Para diferentes tipos de implementação de nuvem existem preocupações específicas.

Aplicações em nuvem trazem novos desafios para a segurança. O gerenciamento de identidades (*Identity Management* - IDM) assume vantagem em toda a área de segurança em nuvem. A computação em nuvem é uma fusão de várias tecnologias para atender às demandas de um ambiente interdependente de software e serviços. Isso significa que IDMs, com tecnologias diferentes, devem interagir e funcionar como uma estrutura transparente. Portanto, IDMs em projetos de nuvens apresenta novas dimensões que IDMs tradicionais não podem atender (GOPALAKRISHNAN, 2009).

A maioria dos fornecedores de nuvem tem uma solução proprietária de IDM, simplificada e com deficiências que devem ser compreendidas. O desafio nesta área é que existem esforços consideráveis para terceirização do IDM, que deu origem ao conceito de identidade como um serviço (*Identity as a Service* - IDaaS) (SPRING, 2009). O foco dos fornecedores de IaaS é em soluções abrangentes interoperáveis e rápidas de implementar.

1.3 TRABALHOS RELACIONADOS

Existem serviços sendo oferecidos baseados em uma terceira parte confiável. Este trabalho visa oferecer um modelo de confiança sem a necessidade de terceiros.

Através da descrição de alguns trabalhos relacionados é possível observar os problemas e desafios nesta área de pesquisa:

No trabalho de (ALBESHRI e CAELLI, 2010) é apresentada uma arquitetura para uma nova abordagem para o problema identificado como Proteção mútua de *Cloud Computing* ("*Mutual Protection for Cloud Computing* (MPCC)"). O conceito principal que fundamenta a MPCC é baseado na filosofia do Controle de Acesso Reverso, onde os consumidores (clientes) controlam e tentam reforçar os meios pelos quais os provedores de nuvem controlam a autenticação e a autorização dentro deste ambiente dinâmico, e o provedor de nuvem garante que a organização do cliente não viola a segurança da estrutura de nuvem global em si. O esquema envolve a correspondência do perfil de segurança do provedor de nuvem com o do cliente, de modo a atingir a aceitação mútua de todo o ambiente de segurança. Este framework ajuda a controlar e monitorar a exigência de que um provedor de nuvem sempre atende aos requisitos de segurança da organização, e que o usuário não pode violar facilmente a postura de segurança do provedor de nuvem, por exemplo, obter acesso aos dados e processos de outro usuário da nuvem.

Em (RANCHAL, BHARGAVA, *et al.*, 2010) é proposta uma abordagem para IDM, que é independente de terceiros (*Trusted Third Party*, TTP) e tem a capacidade de usar dados de identidade em *hosts* não confiáveis. A abordagem é baseada no uso de predicados (atributos) sobre dados criptografados e ambientes distribuídos e heterogêneos de computação para tratar o uso de serviços em nuvem. Além disso, usa "pacote de ativos" (*active bundle*) – que é um agente *middleware* que inclui dados de PII (*Personally Identifiable Information*), políticas de privacidade, uma máquina virtual que reforça as políticas, e tem um conjunto de mecanismos de proteção para proteger-se a si mesmo. Um pacote de ativos interage em nome de um usuário para autenticar serviços em nuvem usando as políticas de privacidade do usuário.

Em (ANGIN, BHARGAVA, *et al.*, 2010) é proposta uma abordagem para IDM em nuvem centrada na entidade. A abordagem é baseada em: (1) pacotes de ativos, cada qual incluindo uma carga de PII, políticas de privacidade e uma máquina virtual que reforça as políticas e usa um conjunto de mecanismos de proteção para se proteger; (2) identificação anônima para mediar as interações entre a entidade e serviços em nuvem usando as políticas de privacidade da entidade. As características principais desta abordagem são: é independente de terceiros, dá informações mínimas para o SP e oferece a capacidade de usar dados de identidades em *hosts* não confiáveis.

1.4 ORGANIZAÇÃO DO TRABALHO

O trabalho está organizado como a seguir:

O capítulo 2 mostra o estado da arte para a computação em nuvem, bem como destaca o serviço EC2 oferecido pela Amazon e, além disso, são apresentadas as questões específicas de segurança para nuvens. No capítulo 3 é apresentado o conceito de gerenciamento de identidades, englobando identidades digitais, SAML e a ferramenta Shibboleth. No capítulo 4 encontra-se a descrição da proposta deste trabalho e, também, o cenário e implementação, desenvolvidos para a realização de testes e obtenção de resultados. O capítulo 5 trata da conclusão do trabalho, bem como são apresentados os desafios para serem tratados em trabalhos futuros e, por último, são apresentadas as referências bibliográficas.

2 COMPUTAÇÃO EM NUVEM

Como uma nova infraestrutura para oferecer serviços, sistemas de computação em nuvem têm muitas vantagens em relação aos serviços oferecidos tradicionalmente, como investimento inicial reduzido, melhoria no desempenho, alta disponibilidade, escalabilidade infinita, grande capacidade de tolerância a falhas, e assim por diante, e consequentemente são acompanhados pela maioria das empresas de TI, como Google, Amazon, Microsoft, Salesforce.com (ZHOU, ZHANG, *et al.*, 2010). A computação em nuvem é um modelo que permite o acesso conveniente à rede sob demanda para um conjunto compartilhado de recursos de computação configuráveis que podem ser rapidamente disponibilizados e liberados com o mínimo esforço de gerenciamento ou interação com o provedor de serviços. Com este modelo de computação sob demanda, as soluções em nuvem são vistas como tendo o potencial para reduzir drasticamente os custos e aumentar a rapidez do desenvolvimento de aplicações (CHANDRAMOULI e MELL, 2010).

2.1 CONCEITOS BÁSICOS

Computação em nuvem é um termo em evolução que descreve o desenvolvimento de muitas das tecnologias e abordagens existentes em computação para algo distinto. A nuvem separa as aplicações e os recursos de informação de sua infraestrutura básica, e os mecanismos utilizados para entregá-los. A nuvem realça a colaboração, agilidade, escalabilidade e disponibilidade, e oferece o potencial para redução de custos através de computação eficiente e otimizada.

Mais especificamente, a nuvem descreve o uso de uma coleção de serviços, aplicações, informação e infraestrutura composta por *pools* de recursos computacionais, de rede, de informação e de armazenamento. Estes componentes podem ser rapidamente organizados, provisionados, implementados, desativados, e escalados, provendo um modelo de alocação e consumo baseado na demanda de recursos.

O *U.S. National Institute of Standards and Technology* (NIST) (MELL e GRANCE, 2011) define Computação em Nuvem descrevendo cinco características essenciais, três modelos de serviço e quatro modelos de implementação. Eles estão sumarizados visualmente na Figura 1 e explicados em detalhes a seguir.

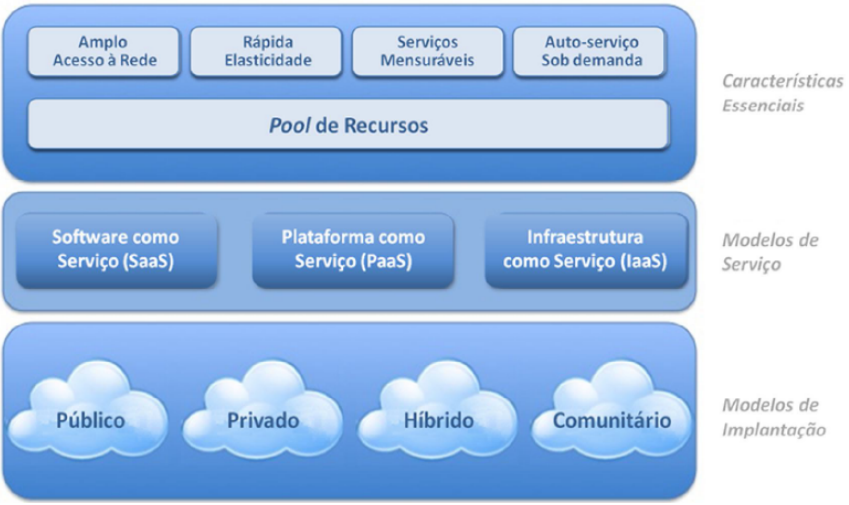

Figura 1. Modelo Visual de Definição de Computação em Nuvem do NIST (MELL e GRANCE, 2011)

2.1.1 Características essenciais de computação em nuvem

Os serviços na nuvem apresentam cinco características essenciais que demonstram suas relações e diferenças das abordagens tradicionais de computação:

No **autoatendimento sob demanda** um consumidor pode unilateralmente provisionar capacidades computacionais como tempo de servidor e armazenamento de rede automaticamente conforme necessário, sem requerer interação humana com o provedor de serviços.

O **amplo acesso à rede** diz respeito às capacidades que estão disponíveis na rede e podem ser acessadas através de mecanismos padrões que promovem o uso por plataformas heterogêneas de clientes leves (*thin clients*) ou não (por exemplo, telefones celulares, laptops, e PDAs) assim como outros serviços de software tradicionais ou baseados em nuvem.

O ***pool* de recursos** refere-se aos recursos de computação do provedor que estão reunidos para servir a múltiplos consumidores usando um modelo multilocação, com diferenças físicas e recursos virtuais dinamicamente atribuídos e reatribuídos de acordo com a demanda do consumidor. Existe um grau de independência de

localização e dessa forma o consumidor geralmente não tem controle ou conhecimento sobre a localização exata dos recursos providos, mas pode ser capaz de especificar a localização em um nível mais alto de abstração (por exemplo, país, estado ou Data Center). Exemplos de recursos incluem armazenamento, processamento, memória, largura de banda, e máquinas virtuais. Até nuvens privadas tendem a reunir recursos entre diferentes partes da mesma organização.

A **elasticidade rápida** tem relação com as capacidades que podem ser rapidamente e elasticamente provisionadas – em alguns casos automaticamente. Para o consumidor, as capacidades disponíveis para o provisionamento geralmente parecem ser ilimitadas e podem ser contratadas em qualquer quantidade e a qualquer hora.

O **serviço medido** é uma característica que permite, aos sistemas em nuvem, automaticamente controlarem e otimizarem o uso de recursos alavancando a capacidade de determinar (medir) em algum nível de abstração apropriado para o tipo de serviço (por exemplo. armazenamento, processamento, largura de banda ou contas de usuário ativas). O uso de recursos pode ser monitorado, controlado e relatado – provendo transparência para ambos o provedor e o consumidor do serviço.

É importante reconhecer que os serviços em nuvem são geralmente, mas nem sempre, utilizados em conjunto com, e habilitado por tecnologias de virtualização. Não existe requisito, no entanto, que relaciona a abstração de recursos com as tecnologias de virtualização e no caso de muitas ofertas, a virtualização por ambientes de sistemas operacionais ou hipervisores não são utilizadas.

2.1.2 Modelos de serviços de nuvem

A entrega de serviços de nuvem é dividida entre três modelos de arquitetura e várias combinações derivadas. As três classificações fundamentais são geralmente referidas como “Modelo SPI”, onde “SPI” significa Software, Plataforma e Infraestrutura (como um Serviço), respectivamente (CSA, 2010b)(CHANDRAMOULI e MELL, 2010):

Software como um Serviço (*Software as a Service, SaaS*). A capacidade oferecida ao consumidor consiste em utilizar as aplicações do provedor executando em uma infraestrutura em nuvem. As aplicações são acessíveis por vários dispositivos através de uma interface simples de cliente como um browser web (exemplo: webmail).

O consumidor não gerencia ou controla a infraestrutura adjacente na nuvem, incluindo rede, servidores, sistemas operacionais, armazenamento, ou nem mesmo as capacidades individuais da aplicação, com a possível exceção de parâmetros limitados de configuração da aplicação específicos para os usuários. Exemplos disso incluem o caso de um provedor em nuvem oferecer uma aplicação usada para uma função específica de negócio, como gestão de relacionamento com o cliente (*Customer Relationship Management, CRM*) ou gestão de recursos humanos (*Human Resources Management, HRM*), baseado em assinatura ou no uso em vez de tradicionalmente baseado na compra ou licenciamento.

Plataforma como um Serviço (*Platform as a Service, PaaS*). A capacidade oferecida ao consumidor é para implementar na infraestrutura em nuvem criada para o usuário ou em aplicações adquiridas usando linguagens de programação e ferramentas suportadas pelo provedor. O consumidor não gerencia ou controla a infraestrutura adjacente na nuvem, incluindo rede, servidores, sistemas operacionais, ou armazenamento, mas tem controle sobre as aplicações implementadas e possivelmente configurações da aplicação referentes ao ambiente do servidor. Exemplos disso incluem o caso de um provedor em nuvem fornecer um conjunto de ferramentas para desenvolvimento e implantação de aplicativos usando diversas linguagens (por exemplo, C, C++, Java), sob um conjunto de framework de aplicações (JEE, .NET, e assim por diante).

Infraestrutura como um Serviço (*Infrastructure as a Service, IaaS*). A capacidade oferecida ao consumidor é de provisionar processamento, armazenamento, redes e outros recursos computacionais fundamentais onde o consumidor está apto a implementar e rodar os softwares que desejar, o que pode incluir sistemas operacionais e aplicações. O consumidor não gerencia ou controla as camadas adjacentes da infraestrutura na nuvem, mas tem controle sobre o sistema operacional, armazenamento, aplicações implementadas e possivelmente controle limitado de componentes específicos de rede (exemplo: firewalls no servidor). Exemplos disso incluem o caso de um provedor em nuvem fornecer hardware físico e virtual (servidores, unidades de armazenamento) para hospedar e ligar todas as aplicações corporativas e armazenar todos os dados da empresa, ou seja, a infraestrutura base para um centro de dados empresarial.

2.1.3 Modelos de implantação de nuvem

Independente do modelo de serviço utilizado (SaaS, PaaS ou IaaS) existem quatro modelos de implantação de serviços de nuvem, com variações para atender a requisitos específicos:

Nuvem Pública. A infraestrutura de nuvem é disponibilizada ao público em geral ou a um grande grupo industrial e é controlada por uma organização que vende os serviços de nuvem.

Nuvem Privada. A infraestrutura da nuvem é operada exclusivamente por uma única organização. Ela pode ser gerida pela organização ou por terceiros, e pode existir no local ou fora do ambiente da empresa.

Nuvem Comunitária. A infraestrutura da nuvem é compartilhada por diversas organizações e suporta uma determinada comunidade que partilha interesses (por exemplo, a missão, os requisitos de segurança, política ou considerações de conformidade). Ela pode ser administrada pelas organizações ou por um terceiro e pode existir no local ou fora do ambiente da empresa.

Nuvem Híbrida. A infraestrutura da nuvem é uma composição de duas ou mais nuvens (privada, comunitária ou pública) que permanecem como entidades únicas, mas estão unidas pela tecnologia padronizada ou proprietária que permite a portabilidade de dados e aplicativos (por exemplo, "*cloud bursting*" para balanceamento de carga entre as nuvens).

2.2 SEGURANÇA PARA COMPUTAÇÃO EM NUVEM

Os controles de segurança para computação em nuvem, em geral, não são diferentes dos controles de segurança para qualquer ambiente de TI. No entanto, em função dos modelos de serviço de nuvem que são empregados, os modelos operacionais e as tecnologias usadas para habilitar tais serviços, a computação em nuvem pode apresentar riscos diferentes para uma organização quando comparada com as soluções tradicionais de TI.

Sob a perspectiva da arquitetura, há muito equívoco em torno de como a nuvem é tanto similar e diferente dos modelos computacionais existentes, e como estas similaridades e diferenças impactam nas abordagens organizacionais, operacionais, e tecnológicas para as práticas de segurança da informação e de redes.

O trabalho de (GROBAUER, WALLOSCHEK e STÖCKER, 2011) enfatiza as questões de segurança na computação em nuvem e a seguir pode ser conferido um texto baseado em tal trabalho.

Frequentemente são evidenciados novos alertas sobre as ameaças e riscos de segurança da computação em nuvem. Na maioria dos casos a segurança é citada como o obstáculo mais significativo para a compreensão da computação em nuvem. Mas este discurso sobre questões de segurança de computação em nuvem torna difícil formular uma avaliação bem fundamentada do impacto de segurança real por dois motivos principais. Em primeiro lugar, em muitas dessas discussões sobre risco, muitas vezes termos de vocabulário básico – incluindo riscos, ameaças e vulnerabilidades – são usados como sinônimos, sem levar em conta suas respectivas definições. Segundo, nem todas as questões levantadas são específicas para a computação em nuvem.

Para alcançar uma compreensão bem fundamentada das características que a computação em nuvem acrescenta no que diz respeito às questões de segurança, devemos analisar como a computação em nuvem influencia questões de segurança estabelecidas. Um fator fundamental neste caso são as vulnerabilidades de segurança: a computação em nuvem torna certas vulnerabilidades bem conhecidas mais significativas, bem como acrescenta outras. Antes de dar uma olhada nas vulnerabilidades específicas da nuvem, no entanto, é preciso primeiro estabelecer o que é uma vulnerabilidade.

De acordo com a taxonomia de risco do Open Group (OPEN GROUP, 2009), vulnerabilidade é a probabilidade de que um ativo será incapaz de resistir às ações de um agente de ameaça. A vulnerabilidade existe quando há uma diferença entre a força a ser aplicada pelo agente de ameaça, e a capacidade de um objeto para resistir a essa força.

A vulnerabilidade é específica da nuvem se ela: (a) é inerente ou predominante nas principais tecnologias da computação em nuvem; (b) tem sua causa principal em uma das características de nuvem essenciais do NIST; (c) é causada quando melhorias na nuvem tornam os controles de segurança consolidados difíceis ou impossíveis de implementar; ou (d) é predominante no atual oferecimento de nuvem.

Vulnerabilidades em relação às principais tecnologias

As principais tecnologias da computação em nuvem – serviços e aplicações web, virtualização, e criptografia – possuem vulnerabilidades que são inerentes à tecnologia ou predominantes nas implementações

atuais da tecnologia. Três exemplos dessas vulnerabilidades são: fuga da virtualização, desvio de sessão e criptografia insegura ou obsoleta.

Em primeiro lugar, a possibilidade de que um invasor pode com sucesso escapar de um ambiente virtualizado (**fuga da virtualização**) está na própria natureza da virtualização. Assim, devemos considerar essa vulnerabilidade como inerente à virtualização e altamente relevante para a computação em nuvem.

Em segundo lugar, as tecnologias de aplicações web devem superar o problema do protocolo HTTP ser um protocolo que não guarda estado, uma vez que as aplicações web necessitam do conceito de estado de sessão. Muitas técnicas implementam tratamento de sessão e muitas implementações de manipulação de sessão são vulneráveis ao desvio de sessão. Sejam as vulnerabilidades de **desvio de sessão** intrínsecas às tecnologias de aplicação web ou apenas predominantes em muitas implementações atuais, essas vulnerabilidades são certamente relevantes para a computação em nuvem.

Por último, os avanços na criptoanálise podem tornar qualquer mecanismo ou algoritmo de criptografia inseguro, conforme novos métodos de quebrá-los são descobertos. É ainda mais comum encontrar falhas cruciais em implementações de algoritmos criptográficos, que pode transformar criptografia forte em criptografia fraca (ou às vezes totalmente sem criptografia). A ampla adoção de computação em nuvem é inconcebível sem o uso de criptografia para proteger a confidencialidade e integridade dos dados na nuvem, portanto, as vulnerabilidades de **criptografia insegura ou obsoleta** são altamente relevantes para computação em nuvem.

Vulnerabilidades em relação às características básicas de nuvem

Conforme descrito anteriormente neste capítulo, o NIST descreve cinco características básicas de nuvem: autoatendimento sob demanda, amplo acesso à rede, pool de recursos, elasticidade rápida e serviço medido. A seguir encontram-se exemplos de vulnerabilidades ocasionadas por uma ou mais destas características:

Acesso não autorizado à interface de gerência: A primeira característica (autoatendimento sob demanda) requer uma interface de gerenciamento, que é acessível aos usuários de serviços em nuvem. O acesso não autorizado à interface de gerenciamento é, portanto, uma vulnerabilidade especialmente relevante para os sistemas em nuvem: a probabilidade de que o acesso não autorizado pode ocorrer é muito

maior do que para os sistemas tradicionais, onde a funcionalidade de gerenciamento é acessível apenas a uns poucos administradores.

Vulnerabilidades do IP: a característica de amplo acesso à rede significa que os serviços em nuvem são acessados através da rede utilizando protocolos padrões. Em muitos casos, esta rede é a Internet, que deve ser considerada não confiável.

Vulnerabilidades de recuperação de dados: as características de pool de recursos e elasticidade rápida implicam que os recursos alocados para um usuário serão realocados para um usuário diferente em um momento posterior. Para os recursos de memória ou de armazenamento, pode, portanto, ser possível recuperar os dados gravados pelo usuário anterior.

Medição e evasão de faturamento: a característica de serviço medido significa que qualquer serviço em nuvem tem uma capacidade de medição em um nível de abstração apropriado para o tipo de serviço (tais como armazenamento, processamento e contas de usuários ativos). A medição de dados é utilizada para otimizar a prestação de serviços, bem como faturamento. Vulnerabilidades relevantes incluem manipulação de medição e dados de faturamento e evasão de faturamento.

Falhas nos meios de segurança conhecidos

As vulnerabilidades nos meios de segurança padrões devem ser consideradas específicas de nuvem se as inovações de nuvens provocam dificuldades na implementação dos controles. Tais vulnerabilidades são também conhecidas como desafios de controle.

Aqui serão tratados três exemplos desses desafios de controle. Primeiro, redes virtualizadas oferecem controles baseados em redes insuficientes. Dada a natureza dos serviços em nuvem, o acesso administrativo para a infraestrutura de rede IaaS e da capacidade de infraestrutura de rede sob medida são normalmente limitados, portanto, os controles padrões, como o zoneamento de rede baseado em IP, não podem ser aplicados. Além disso, as técnicas padrão, tais como exploração de vulnerabilidade baseada em rede normalmente é vedada por fornecedores de IaaS, pois as verificações amigáveis não podem ser distinguidas de atividades de invasores. Finalmente, as tecnologias como a virtualização significa que o tráfego de rede ocorre tanto nas redes reais como nas virtuais, como quando dois ambientes de máquinas virtuais (VMEs) hospedados no mesmo servidor se comunicam. Tais questões constituem um desafio de controle, pois controles de segurança

comprovados em nível de rede podem não funcionar em um determinado ambiente de nuvem.

O segundo desafio consiste no mau procedimento de gerenciamento de chaves. A infraestrutura de computação em nuvem requer gerenciamento e armazenamento de vários tipos de chaves. Como máquinas virtuais não tem uma infraestrutura de hardware fixa e o conteúdo baseado em nuvem normalmente está distribuído geograficamente, é mais difícil aplicar controles padrões para chaves em infraestruturas de nuvem.

Por fim, as métricas de segurança não são adaptadas para as infraestruturas em nuvem. Atualmente não há métricas de segurança específicas de nuvens padronizadas que os clientes da nuvem possam usar para monitorar o status da segurança de seus recursos na nuvem. Até que as métricas de segurança padrão sejam desenvolvidas e implementadas, os controles para avaliação, auditoria e responsabilidade de segurança são mais difíceis e custosos, e podem até mesmo ser impossíveis de empregar.

Vulnerabilidades predominantes no atual oferecimento de nuvem

Se a vulnerabilidade é predominante no atual oferecimento de nuvem, ela deve ser considerada como específica de nuvem. Por exemplo, vulnerabilidades de injeção e esquemas de autenticação fracos.

Vulnerabilidades de injeção são exploradas através da manipulação de entradas de serviço ou aplicação para interpretar e executar partes deles contra o objetivo do programador. Exemplos de vulnerabilidades de injeção incluem:

Injeção de SQL, em que a entrada contém código SQL que é executado erroneamente no back-end do banco de dados;

Injeção de comando, em que a entrada contém comandos que são executados erroneamente através do sistema operacional; e

Cross Site Scripting (XSS), onde a entrada contém código *JavaScript* que é executado erroneamente pelo navegador da vítima. Além disso, muitos mecanismos de autenticação usados são fracos. Por exemplo, nomes de usuários e senhas para autenticação são fracos, por causa do comportamento inseguro do usuário (escolhendo senhas fracas, reutilização de senhas, e assim por diante), e também pelo fato de haver limitações inerentes a um mecanismo de autenticação de fator único.

Além disso, a implementação dos mecanismos de autenticação pode ter falhas e permitir, por exemplo, interceptação e repetição de

credenciais. A maioria das aplicações web nos atuais serviços de nuvem empregam nomes de usuários e senhas como mecanismo de autenticação.

2.3 SERVIÇO DE COMPUTAÇÃO EM NUVEM AMAZON EC2

O Amazon EC2 (*Elastic Compute Cloud*) (AMAZON, 2011) é um serviço web que oferece capacidades computacionais com características de escalabilidade, disponibilidade, elasticidade e alto desempenho para aplicações executadas neste ambiente. São disponibilizados como instâncias de servidores em centros de dados da Amazon e usados para construir e hospedar sistemas de software. Os recursos de infraestrutura que o EC2 fornece podem ser acessados por meio de APIs, ou ferramentas da web e utilitários.

O EC2 disponibiliza uma infraestrutura completa para computação em diversos níveis de processamento, desde tarefas simples até de alto desempenho e possui uma gerência eficaz dos recursos. Do ponto de vista de economia, o EC2 reduz os custos, através da computação sob demanda, otimizando os recursos computacionais além de fornecer aos desenvolvedores ferramentas para construir aplicações escaláveis. Com o EC2 é possível usar e pagar somente pela estrutura necessária. Isso elimina a necessidade de fazer compras de hardware grandes e caras, reduz a necessidade de previsão de tráfego, e possibilita que, automaticamente, os recursos de TI sejam redimensionados para lidar com as mudanças nos requisitos e pontos importantes relacionados à aplicação ou serviço.

O EC2 permite um controle completo de suas instâncias, sendo possível acessar e interagir com cada umas destas, de forma similar a máquinas convencionais. Também é possível escolher as características de cada instância, tais como sistema operacional, pacotes de softwares e as configurações das máquinas, como CPU, memória e armazenamento. Para garantir a segurança, o EC2 utiliza firewall para controlar o acesso às instâncias, criando ambientes virtuais privados. Para utilizar o EC2, primeiro é necessário criar uma imagem de máquina para executar as aplicações, chamada de AMI (*Amazon Machine Image*), que contém os aplicativos, bibliotecas, dados e configurações associadas. Esta imagem é armazenada em um repositório seguro, rápido e confiável.

Uma *Amazon Machine Image* (AMI) é um modelo que contém uma configuração de software (por exemplo, sistema operacional,

servidor de aplicação e aplicações). A partir de uma AMI, é possível iniciar instâncias que estão executando cópias da AMI. Várias instâncias podem ser iniciadas, conforme mostrado na Figura 2.

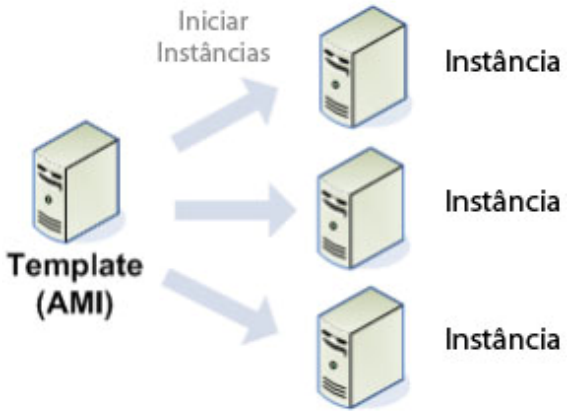

Figura 2. *Amazon Machine Image* e suas respectivas instâncias.

As instâncias permanecem executando até que sejam paradas ou finalizadas, ou até que haja uma falha. Se uma instância falhar, uma nova poderá ser iniciada a partir da AMI.

Um cliente pode usar uma ou várias AMIs. A partir de uma única AMI, é possível iniciar diferentes tipos de instâncias. Como ilustrado na Figura 3, é possível selecionar um tipo de instância específica com base na quantidade de memória ou processamento necessário para o aplicativo ou software que será executado na instância.

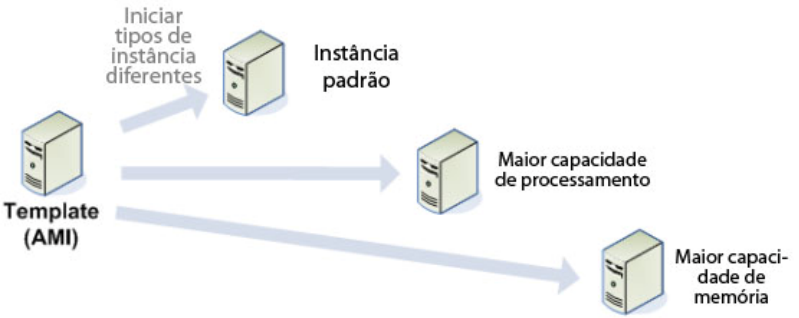

Figura 3. AMI com diferentes tipos de instâncias.

A Amazon tem centros de dados em áreas diferentes do mundo (por exemplo, América do Norte, Europa, Ásia, etc.) Do mesmo modo, o EC2 está disponível para uso em regiões diferentes. Ao iniciar instâncias em regiões separadas, um aplicativo pode ser projetado para estar mais perto de clientes específicos, para atender aos requisitos legais ou outros. Os preços para o uso do Amazon EC2 variam por região.

Cada região contém várias localizações distintas chamadas de zonas de disponibilidade (ilustrado na Figura 4). Cada zona de disponibilidade é projetada para ser isolada de falhas em outras zonas de disponibilidade e para fornecer conectividade de rede barata e de baixa latência para outras zonas da mesma região. Ao iniciar instâncias em zonas de disponibilidade separadas, é possível proteger as aplicações da falha de um único local.

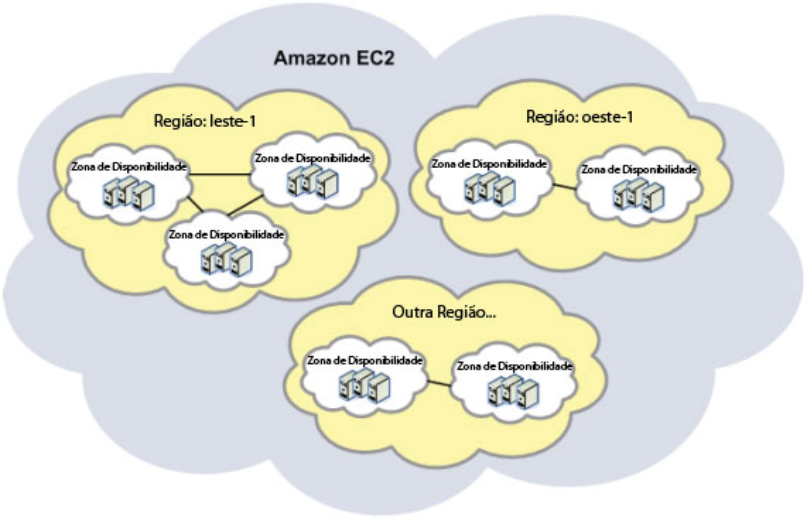

Figura 4. Regiões e zonas de disponibilidade.

Os dois tipos de armazenamento mais comumente usados em conjunto com o EC2 são: *Amazon Simple Storage Service* (Amazon S3) e *Amazon Elastic Block Store* (Amazon EBS).

O Amazon S3 é o armazenamento para a Internet. Ele fornece uma interface de serviço web simples que permite armazenar e recuperar qualquer quantidade de dados de qualquer lugar na web. O S3 é um sistema de arquivos distribuído, utilizado para recuperar e armazenar

dados. O S3 fornece um repositório seguro, confiável e rápido para armazenar as imagens AMI. E ainda armazena e recupera os resultados intermediários durante a execução das tarefas de processamento. Durante a execução, cada tarefa busca o arquivo no S3 e faz o devido processamento. As soluções EC2 armazenam arquivos como objetos no S3.

O Amazon EBS fornece a persistência dos dados e agem como discos rígidos das máquinas (Figura 5). É possível criar imagens (*snapshots*) desses volumes, para posterior recuperação de dados. Essas imagens podem ser usadas para serem usadas em outras instâncias. Além disso, cada instância pode possuir mais de um volume EBS.

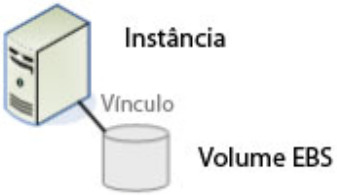

Figura 5. Amazon EBS.

3 GERENCIAMENTO DE IDENTIDADES

Este capítulo apresenta os conceitos básicos que envolvem o tema gerenciamento de identidades, como identidade digital e sistemas de gerenciamento de identidades. Além disso apresenta o padrão SAML (*Security Assertion Markup Language*), juntamente com sua arquitetura e modo de funcionamento e por último apresenta a ferramenta Shibboleth, que é baseada no padrão SAML.

A tecnologia digital de gerenciamento de identidade é fundamental para personalizar e melhorar a experiência do usuário, protegendo a privacidade, sustentando a responsabilidade nas transações e interações, e em conformidade com os controles regulamentares. Uma identidade digital pode ser definida como a representação digital das informações conhecidas sobre um indivíduo ou organização específica. Essas informações podem ser usadas para fins diferentes, que vão desde permitir que se possa provar a sua declaração de identidade (como a utilização de certidão de nascimento ou passaporte) até permissões de confirmação (como o uso de uma carteira de motorista para confirmar o direito de dirigir). Uma identidade digital pode incluir informações atribuíveis a um indivíduo, tais como nome, número de seguro social (SSN), ou número do passaporte. Além disso, também pode incorporar informações biométricas, como a íris ou impressões digitais e informações sobre as atividades do usuário, incluindo as pesquisas na web e transações virtuais. Uma identidade digital pode englobar também identificadores, como nomes de usuário e pseudônimos, usados por indivíduos quando interagem com os sistemas computacionais ou com outros indivíduos no mundo virtual (BERTINO e TAKAHASHI, 2011).

Segundo (CHADWICK, 2009), a identidade é a “Representação de uma entidade (ou grupo de entidades) sob a forma de um ou mais elementos de informação (atributos) que possibilitam a entidade ser reconhecida unicamente dentro de um contexto”. Entretanto, esta definição é muito geral e pode representar qualquer objeto. Então, pode-se restringir para o gerenciamento de identidades de pessoas em vez de qualquer objeto. Neste contexto, os elementos de informação são restritos às informações de identificação pessoal (*Personally Identifiable Information*, PII), que são as informações capazes de identificar uma determinada pessoa, como por exemplo, cor do cabelo, som da voz,

altura, nome, qualificações, ações passadas, reputação, registros médicos, entre outros.

O gerenciamento de identidades é um processo no qual cada pessoa ou recurso é fornecido com um identificador único, utilizados para identificar essa entidade única. A gestão de identidade é utilizada para controlar o acesso a qualquer sistema/recurso através dos direitos associados ao usuário e as restrições estabelecidas para a identidade (BELAPURKAR, CHAKRABARTI, *et al.*, 2009).

Segundo (JøSANG e POPE, 2005), o gerenciamento de identidades (*Identity Management*, ou IdM) é uma combinação de práticas e tecnologias para representação e reconhecimento de entidades como identidades digitais. Sendo que do ponto de vista dos provedores de serviços, IdM representa uma atividade atribuída ao provedor de serviços para gerenciar as identidades dos usuários dos serviços. Em (CHADWICK, 2009) o gerenciamento de identidades é definido como sendo um conjunto de funções e capacidades, como administração, gerência e manutenção, descoberta, troca de informação, aplicação de políticas e autenticação, usadas para garantir as informações da identidade, garantindo dessa maneira a segurança.

O gerenciamento de identidade não trata apenas de tecnologia, mas compreende três elementos indispensáveis: diretivas, processos e tecnologias. As diretivas se referem aos limites e padrões que precisam ser seguidos para cumprir as normas e as práticas comerciais recomendadas; os processos descrevem as sequências de etapas que levam à conclusão de funções ou tarefas comerciais; as tecnologias são as ferramentas automatizadas que ajudam a atingir objetivos comerciais de forma mais eficiente e precisa, ao mesmo tempo em que se respeitam os limites e as orientações especificadas nas diretivas (SANTOS, 2008).

3.1 SISTEMAS DE GERENCIAMENTO DE IDENTIDADES

Conforme (BELAPURKAR, CHAKRABARTI, *et al.*, 2009), um sistema de gerenciamento de identidades provê ferramentas para o gerenciamento das identidades individuais em um mundo digital. Em nosso dia-a-dia temos como exemplo de sistema de gerenciamento de identidades a carteira de motorista, a carteira de identidade ou passaporte. Uma pessoa portando um passaporte pode entrar em um país devido aos direitos relacionados à identidade. O passaporte dá-lhes

também o direito de acesso aos recursos do país, bem como o direito de executar determinadas operações, como votar.

Os sistemas de gerenciamento de identidades são importantes no atual contexto empresarial para gerir um grande número de usuários. Estes sistemas automatizam um grande número de tarefas, como a sincronização de senha, criação e/ou eliminação de identidades de usuários, redefinição de senha e gestão global do ciclo de vida da identidade dos usuários. Uma das principais vantagens desses sistemas é que, quando novas aplicações são fornecidas em uma empresa, eles podem aproveitar os dados do sistema de gerenciamento de identidades, sem a necessidade de dados específicos para a nova aplicação.

Algumas funcionalidades especializadas de sistemas de gerenciamento de identidades inclui um acesso único (*Single sign-on*, SSO), onde um usuário não precisa fazer *login* várias vezes ao chamar várias aplicações e pode reutilizar o estado autenticado de um pedido anterior, na mesma sessão. SSO é fundamental para uso da empresa de soluções de gerenciamento de identidades.

Um sistema de gerenciamento de identidades é composto de protocolos e componentes de software que tratam as identidades dos indivíduos durante todo o ciclo de vida de suas identidades. Um sistema de gerenciamento de identidades envolve três principais tipos de entidades, a saber: o usuário, o Provedor de Identidades (*Identity Provider*, ou IdP) e o Provedor de Serviços (*Service Provider*, SP). Os IdPs são responsáveis por emitir e gerenciar as identidades de usuários e emitir credenciais. Os provedores de serviços (também conhecidos como partes confiantes) são entidades que fornecem serviços aos usuários baseado em suas identidades (atributos) (BHARGAV-SPANTZEL, CAMENISCH, *et al.*, 2007).

3.1.1 Requisitos de um sistema de gerenciamento de identidades.

(ICPP, ULD e SNG, 2003) destaca os seguintes requisitos necessários para um sistema de gerenciamento de identidades:

Funcionalidade. O sistema de gerenciamento de identidades auxilia o usuário a controlar sua identidade. Neste sentido, (DAMIANI, VIMERCATI e SAMARATI, 2003) refere-se ao mecanismo para revogação de identidades, o qual permitirá o sistema prover uma forma para que seus usuários possam gerenciar as informações contidas em suas identidades bem como revogá-las. Além disso, o usuário deve ser

informado sobre o contexto de uma situação, oferecendo-lhe escolha, caso necessário.

Usabilidade. A interface com o usuário precisa permitir que este interprete o contexto e escolha a identidade desejada; gerencie a identidade em um mundo digital; e trabalhe com diferentes dispositivos, tais como PCs (*Personal Computer*), PDAs (*Personal Digital Assistant*) e *SmartPhones*.

Segurança. Um IMS deve ser eficiente contra os ataques contra disponibilidade, integridade e confidencialidade de seus serviços e informações. Isto se torna necessário devido ao grande volume de informações confidenciais existentes dos usuários, que pode causar tentativas de espionagem, manipulação e roubo de identidades.

Privacidade. Os usuários devem possuir meios para que possam expressar, e fazer valer, suas preferências de privacidade sobre as informações pessoais presentes em suas identidades.

Anonimato. Deve-se garantir aos usuários o direito de permanecerem anônimos de forma que as informações fornecidas com sua identidade digital não possam ser usadas para descobrir dados de suas outras identidades. O uso de pseudônimos é uma forma para garantir o anonimato.

Aplicação da lei. As partes responsáveis estão geralmente interessadas em coletar o máximo de informações para dar evidências e tornar os procedimentos criminais mais fáceis e efetivos.

Fidelidade. A fidelidade é um pré-requisito para todas as transações onde o usuário confie no provedor de serviço, até mesmo nos casos onde o usuário tem pleno domínio sobre o hardware, software e fluxo de dados.

Gerenciamento de confiança. Relações de confiança entre provedores de serviços e provedores de identidades de diferentes domínios permitem que as identidades emitidas em um domínio sejam aceitas em outro. É necessário prover uma forma para indicar o nível de confiança associado a cada relação, o que irá influenciar no comportamento dos provedores de serviço.

Interoperabilidade. Uma aplicação de gerenciamento de identidades deve implementar interfaces compatíveis com padrões internacionais, além de oferecer compatibilidade e integração com sistemas existentes. Segundo (DAMIANI, VIMERCATI e SAMARATI, 2003) as identidades dos usuários devem ser representadas em um formato comum para que possam ser compreendidas e validadas em diferentes domínios administrativos e de segurança

3.1.2 As sete leis para que um sistema de gerenciamento de identidades tenha sucesso.

Como visto em (CHADWICK, 2009) depois do insucesso do sistema de gerenciamento de identidades Passport da Microsoft, Kim Cameron (CAMERON, 2005) investigou sobre o que é necessário para construir tais sistemas com sucesso. Um dos resultados finais foram as 7 leis da identidade, descritas a seguir.

1. Consentimento e controle do usuário. Um sistema de identidades deve revelar informações de identificação de um usuário apenas com seu consentimento. Acredita-se que os usuários deixam de confiar em um sistema que expõe seus atributos a outros, sem uma permissão explícita. Um usuário precisa ter certeza que todo sistema irá proteger seus atributos e respeitar a forma como devem ser usados.

2. Revelação mínima para uso restrito. A solução que expõe a menor quantidade de informações de identificação e melhor limita seu uso é a solução mais estável a longo prazo. Todos os sistemas são vulneráveis a ataques e roubo (ou perda) de informações confidenciais. Portanto, os sistemas devem diminuir as informações trocadas e excluí-las tão logo quanto possível.

3. Justificação das partes. Sistemas de gerenciamento de identidades devem ser projetados para revelar informações sobre as identidades somente para as partes que justifiquem tal necessidade. Somente as partes envolvidas diretamente na transação com o usuário deverão conhecer suas identidades.

4. Identidade direcionada. Um sistema de identidades deve dar suporte tanto para identificadores “onidirecionais” para uso de entidades públicas quanto para identificadores “unidirecionais” para uso de entidades privadas. Dessa forma, um usuário poderia escolher quais identificadores usar de acordo com cada provedor de serviços que for interagir, garantindo seu anonimato, mesmo que provedores de serviços entrem em conluio.

5. Pluralismo de tecnologias e operadores. Um sistema de identidades deve possibilitar o funcionamento entre tecnologias diferentes usadas pelos diversos provedores de identidades. É necessário que exista um meta identificador aliado a um protocolo comum para o transporte das identidades.

6. Integração com o usuário. O usuário deve ser compreendido como parte integrante do sistema para assim se proteger contra ataques

de roubo de identidade. A interação humana com os sistemas é o elo mais fraco e a segurança nessa comunicação se torna essencial.

7. Experiência consistente através de contextos. A unificação dos meta sistemas de identificação deve garantir aos usuários uma experiência de uso simples e consistente, mesmo atravessando contextos com diferentes tecnologias e operadores.

3.1.3 Funções dos sistemas de gerenciamento de identidades

A seguir são descritas as principais funções de um sistema de gerenciamento de identidades (BELAPURKAR, CHAKRABARTI, *et al.*, 2009)(BHARGAV-SPANTZEL, CAMENISCH, *et al.*, 2007)(CSA, 2010b)(SANTOS, 2008):

A prática de **provisionamento** de identidades dentro de uma organização trata da concessão (provisionamento) e revogação (deprovisionamento) de vários tipos de contas de usuários (por exemplo, usuário final, administrador da aplicação, administrador de TI, supervisor, desenvolvedor, administrador de faturamento) para utilização de serviços.

A **autenticação** é o ato do usuário se identificar utilizando-se de mecanismos diversos, como *login*, senha, biometria, *token*, entre outros. Pode ser utilizada para acesso lógico (computadores e sistemas) ou físico (catracas e portas). A autenticação básica em sistemas é baseada em identificador e senha. Outros fatores que podem controlar a autenticação são horários permitidos e localidades permitidas (computadores autorizados, catracas e portas) (SANTOS, 2008). O processo de garantir que o indivíduo é realmente quem afirma ser é crucial em qualquer cenário de computação. Este processo é chamado de autenticação. A autenticação é um passo fundamental, necessário antes de permitir qualquer pessoa (entidade) realizar uma operação em um computador ou acesso a qualquer parte de um sistema. O administrador interage com o sistema fornecendo uma chave ou uma informação incompleta que somente ele sabe ou é capaz de gerar.

Uma necessidade comum em segurança é proporcionar diferentes níveis de acesso (por exemplo negar / permitir) para diferentes partes ou operações em um sistema de computação. Esta necessidade é denominada **autorização**. O tipo de acesso é definido pela identidade da pessoa, e pelo tipo de operação ou parte do sistema requisitado. O controle de acesso pode ser feito de várias formas, incluindo controle de

acesso obrigatório, controle de acesso baseado em papéis e controle de acesso discricionário.

Uma **federação** é um grupo de organizações ou provedores de serviços que estabelecem um círculo de confiança e permite o compartilhamento de informações de identidades de usuários entre si. Esta característica será retomada mais à frente e explicada em detalhes.

3.1.4 Modelos de sistemas de gerenciamento de identidades

Em (JøSANG e POPE, 2005)(JøSANG, FABRE, *et al.*, 2005)(BHARGAV-SPANTZEL, CAMENISCH, *et al.*, 2007)(AHN e LAM, 2005)(LEE, JEUN e JUNG, 2009) os modelos são divididos em: modelo tradicional (ou isolado, ou silo), modelo centralizado, modelo federado (ou distribuído) e modelo centrado no usuário. A seguir cada modelo é apresentado detalhadamente

Modelo Silo

Na maioria dos modelos de gerenciamento de identidades, o provedor fornece serviços web e tem a função de IdP que emite um ID para uso do serviço web. O provedor de serviços especifica o ID para o usuário que é válido apenas dentro do seu domínio de serviço. Isto é, o usuário recebe um ID diferente para cada provedor de serviço. Este tipo de modelo IdM é chamado Silo, como descrito na Figura 6.

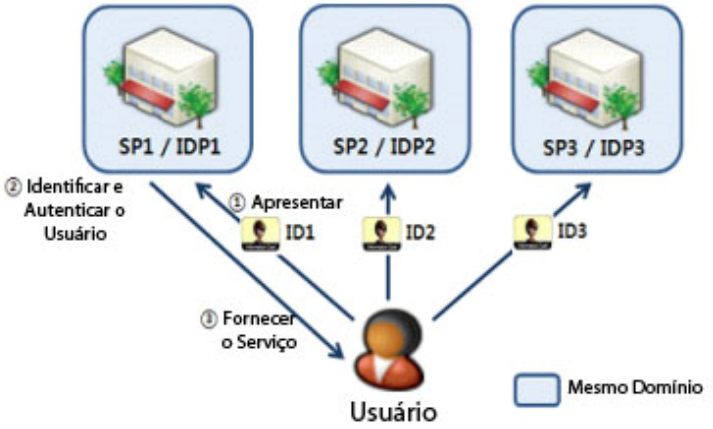

Figura 6. Modelo Silo **(LEE, JEUN e JUNG, 2009)**

Este modelo é o mais simples e fácil de ser implementado. Entretanto, se o número de serviços online aumenta, o número de ID que deve ser gerenciado pelo usuário também aumenta. Além disso, as informações pessoais de identificação (PII) submetidas a muitos provedores de serviços para a criação de ID é distribuído e gerenciado por cada provedor de serviços. Como resultado, se a informação é revelada por algum provedor com segurança vulnerável, pode causar sérias ameaças na violação da privacidade.

Modelo Centralizado

O modelo centralizado permite o usuário se conectar a todos os provedores de serviço dentro do mesmo domínio usando um ID emitido por um único IdP, como por exemplo, o Microsoft Passport. O modelo centralizado pode ser implementado de diferentes maneiras e o tipo mais simples é o Single Sign On (SSO). A Figura 7 descreve o processo de uso de um ID com Single Sign On.

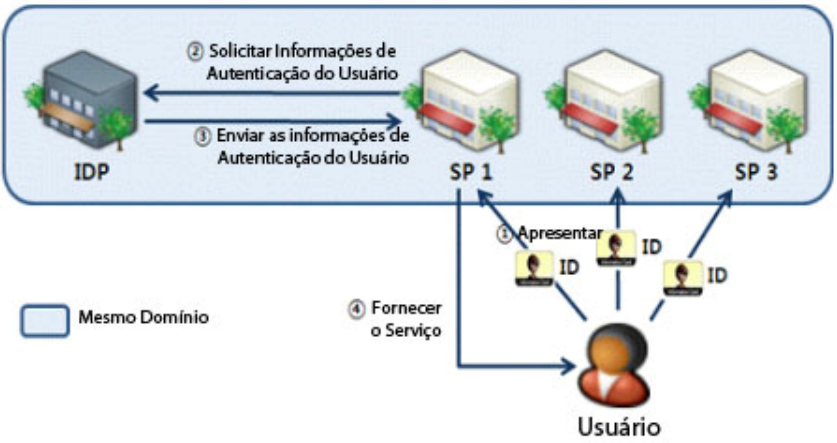

Figura 7. Modelo Centralizado **(LEE, JEUN e JUNG, 2009)**.

Neste modelo, um usuário pode facilmente acessar e usar vários serviços web com um único ID, mas este modelo tem algumas vulnerabilidades também. Se o atacante conseguir a informação de autenticação do único ID do usuário, poderá facilmente se passar pelo usuário para acessar todos os serviços no domínio.

Modelo Federado

No modelo federado, o IdP compartilha o ID do usuário entre provedores de serviços pertencentes a um círculo de confiança. Este círculo de confiança é constituído através do acordo, referente a segurança e autenticação, entre os IdPs e provedores de serviços, e são federados por este acordo. A Figura 8 descreve o modelo Federado.

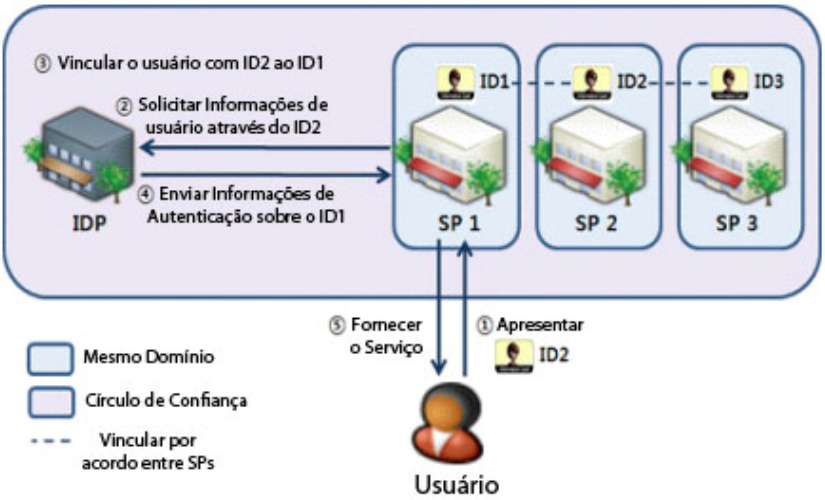

Figura 8. Modelo Federado **(LEE, JEUN e JUNG, 2009)**.

Este modelo pode oferecer serviço SSO em ambiente aberto e pode facilmente ser compatível com o modelo centralizado. Entretanto, é difícil diferenciar entre um usuário real e um provedor de serviços mascarado como usuário. Portanto, neste modelo são necessários recursos técnicos mais avançados para fazer esta distinção. Também, se os provedores de serviços forem maliciosos, podem investigar o usuário mapeando vários IDs do usuário porque uma parte das informações (PII) é compartilhada entre IdPs e SPs, podendo pôr em risco a privacidade do usuário. Por isso, neste modelo é obrigatório o consentimento do usuário em relação ao tipo, escopo e propósito de uso de suas informações, submetidas durante o mapeamento entre o IdP e o SP.

Modelo Centrado no Usuário

Neste modelo o usuário controla e gerencia suas informações (PII) e políticas de uso de ID, como descrito na Figura 9. O usuário pode controlar todos os passos exigidos para criar, usar e manter suas PII e IDs. Especialmente, este modelo requer o consentimento do usuário explicitamente, antes de transmitir, compartilhar suas PII e IDs, bem como suas informações de autenticação.

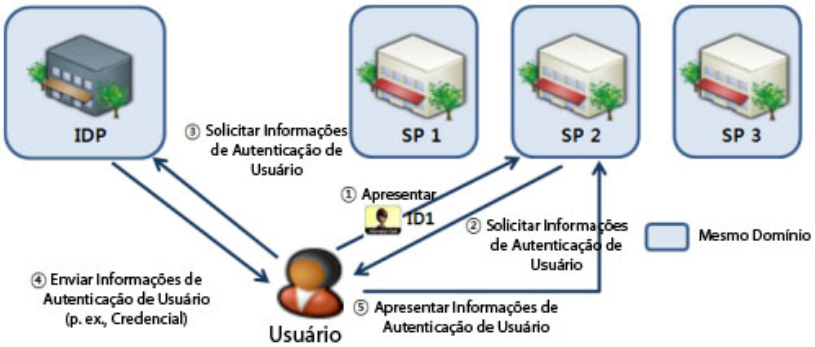

Figura 9. Modelo Centrado no Usuário **(LEE, JEUN e JUNG, 2009)**.

Comparado aos outros modelos, este pode proteger as informações de identidade com maior segurança.

3.2 SAML

SAML (*Security Assertion Markup Language*) é uma linguagem de marcação baseada em XML para troca de informações de segurança entre diferentes domínios. É um padrão definido pela OASIS (*Organization for the Advancement of Structured Information Standards*). Estas informações de segurança são representadas em forma de asserções SAML portáveis, onde as aplicações que funcionam em um determinado domínio podem confiar. O padrão SAML define regras e uma sintaxe precisa para requisição, criação, comunicação e uso dessas asserções SAML. O Comitê Técnico de Serviços de Segurança (*Security Services Technical Committee*, SSTC) da OASIS é o órgão responsável pelo desenvolvimento e suporte do padrão SAML.

A SAML permite que os membros de uma federação criem e transmitam asserções sobre entidades, que são usadas para permitir o acesso a diferentes serviços. Foi projetado tendo em vista segurança e extensibilidade e é reconhecido como uma das melhores soluções para federação de identidades.

Entre as possibilidades do uso de SAML estão o estabelecimento de transações de alto nível entre participantes, autenticação iniciada pelo provedor de identidade ou pelo provedor de serviços, autorização baseada em atributos, *Single Sign-off* e administração centralizada de identidades.

A SAML é hoje um padrão de fato e é utilizada, em parte ou por completo, em muitos outros projetos como Shibboleth (SCAVO e CANTOR, 2005), *Liberty Alliance* (LIBERTY ALLIANCE, 2003), *WS-Federation* (OASIS, 2009), *OpenID* (OPENID, 2010) e *CardSpace* (CHAPPELL, 2006).

3.2.1 Especificações SAML

Nesta seção é descrita a arquitetura da SAML e seu funcionamento, o que se torna necessário para posteriormente compreender o funcionamento da ferramenta Shibboleth, baseada em SAML.

A OASIS lançou um conjunto de especificações para definir uma infraestrutura para troca dinâmica de informações de segurança entre parceiros de negócio. A SAML representa informações de segurança na forma de asserções, bem como protocolos para requisição e envio dessas asserções (OASIS, 2005).

Nas versões iniciais (1.0 e 1.1), a SAML tinha o objetivo de facilitar a troca de informações de identidade e autorização de usuários para permitir autenticação única (*Single Sign-On* - SSO) na web. Já a versão 2.0 tem objetivos mais amplos, como a formação de federações para compartilhamento de informações de segurança e gerenciamento de identidades federadas.

Como pode ser visto na Figura 10 a SAML consiste de blocos de construção de componentes que quando combinados suportam diversos cenários. Os componentes permitem principalmente a transferência de informações de identidades, autenticação, atributos e autorização entre organizações autônomas que tem um relacionamento de confiança estabelecido. O núcleo da especificação SAML define a estrutura e o

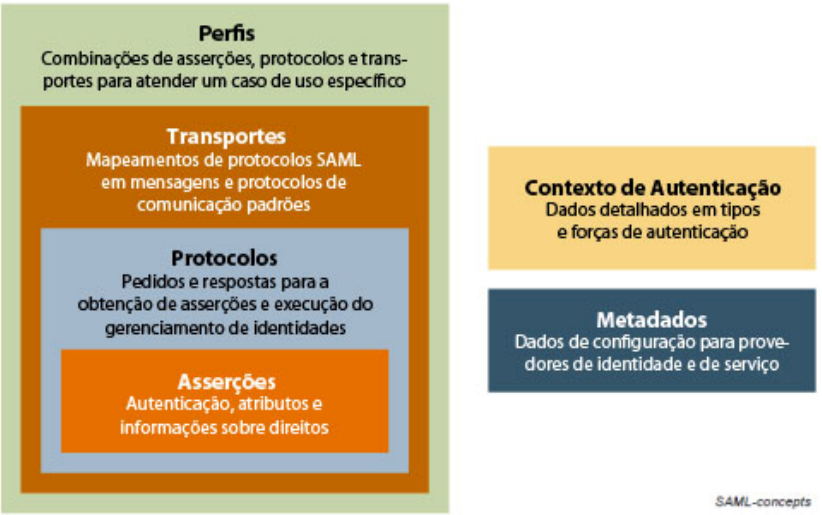

Figura 10. Arquitetura da SAML (OASIS, 2008)

conteúdo das mensagens, tanto das asserções como dos protocolos, usados para transferir estas informações.

As asserções SAML transportam declarações sobre uma entidade (principal) que uma parte declarante (*asserting party*) afirma ser verdadeira. O conteúdo e a estrutura válidos de uma asserção são definidos pelo *XML schema* da asserção SAML. As asserções normalmente são criadas por uma parte declarante baseadas em um pedido de algum tipo de uma terceira parte confiável (ou parte confiante - *relying party*), embora sob algumas circunstâncias, as asserções possam ser passadas a uma parte confiante de maneira espontânea. As mensagens do protocolo SAML são usadas para realizar os pedidos definidos em SAML e retornar as respostas apropriadas. A estrutura e o conteúdo destas mensagens são definidos pelo *XML Schema* do protocolo definido para SAML.

Os meios pelos quais a comunicação de baixo nível ou os protocolos de mensagens (como HTTP ou SOAP) são usados para transportar mensagens do protocolo SAML entre participantes é definido pelos *bindings* SAML (SAML *bindings*), que serão chamados daqui por diante de *transporte*.

Os perfis SAML são definidos para satisfazer um caso de uso específico de negócios, por exemplo, o perfil de SSO do navegador web. Os perfis tipicamente definem restrições nos conteúdos de asserções

SAML, protocolos e transporte para resolver o caso de uso do negócio de modo interoperável. Há também perfis de atributos (*Attribute Profiles*), que não se referem a nenhum protocolo de mensagens e transporte, que definem como a troca de informações de atributo usando asserções de forma que se alinham com uma série de ambientes de uso comum (por exemplo, diretórios X.500/LDAP, DCE).

Outros dois conceitos sobre SAML são úteis para a criação e implantação de um ambiente SAML, são eles:

Metadados. Define um modo de expressar e compartilhar informações de configuração entre partes SAML. Por exemplo, o transporte suportado de uma entidade, papéis operacionais (IDP, SP, etc), informações sobre o identificador, suporte a atributos de identidade, e a informação principal para codificação e assinatura podem ser expressos usando documentos XML de metadados SAML. São definidos por um *XML Schema* próprio.

Contexto de autenticação. Em várias situações, um provedor de serviços pode precisar de informações detalhadas em relação ao tipo e à força da autenticação que um usuário utilizou ao se autenticar em um provedor de identidades. Um contexto de autenticação (*authentication context)* é usado em (ou referido a) uma instrução de autenticação de asserção para transmitir esta informação. Um SP também pode compreender um contexto de autenticação em um pedido de um IdP para solicitar que o usuário seja autenticado usando um conjunto de requisitos de autenticação específicos. Existe um XML *Schema* que define os mecanismos para criação das declarações de contexto de autenticação e um conjunto de Classes *Authentication Context* definidas em SAML, cada qual com seu próprio *XML Schema*, bem como diretrizes para a concepção personalizada de novos perfis e transporte de forma a garantir a máxima interoperabilidade.

A seguir são apresentados todos os componentes da arquitetura em maiores detalhes.

Asserções

As asserções SAML especificam o formato para representar informações de segurança sobre um sujeito, que pode ser uma pessoa, organização ou computador. Por exemplo, uma asserção SAML pode determinar (declarar) que o sujeito se chama “John Doe” tem o endereço de email “john.doe@example.com”, e é um membro do grupo “*engineering*”. Essas informações são atestadas por uma entidade

```
1: <saml:Assertion
xmlns:saml=”urn:oasis:names:tc:SAML:2.0:assertion”
2: Version="2.0"
3: IssueInstant="2005-01-31T12:00:00Z">
4: <saml:Issuer Format=urn:oasis:names:SAML:2.0:nameid-
format:entity>
5: http://idp.example.org
6: </saml:Issuer>
7: <saml:Subject>
8: <saml:NameID
9: Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">
10: j.doe@example.com
11: </saml:NameID>
12: </saml:Subject>
13: <saml:Conditions
14: NotBefore="2005-01-31T12:00:00Z"
15: NotOnOrAfter="2005-01-31T12:10:00Z">
16: </saml:Conditions>
17: <saml:AuthnStatement
18: AuthnInstant="2005-01-31T12:00:00Z"
SessionIndex="67775277772">
19: <saml:AuthnContext>
20: <saml:AuthnContextClassRef>
21:
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTran
sport
22: </saml:AuthnContextClassRef>
23: </saml:AuthnContext>
24: </saml:AuthnStatement>
25: </saml:Assertion>
```

Figura 11. Exemplo de asserção SAML **(OASIS, 2008)**

denominada de parte declarante (*asserting party*) ou autoridade SAML (*SAML authority*).

A Figura 11 mostra um fragmento XML contendo um exemplo de asserção SAML com uma única declaração de autenticação, que contém as seguintes informações:

Linha 1: começa a asserção e contém a declaração do espaço de nomes da asserção SAML que por convenção é representado nas especificações com o prefixo saml:.

Linhas 2 a 6 fornecem informação sobre a natureza da asserção: qual a versão da SAML está sendo usada, quando a asserção foi criada e quem a emitiu.

Linhas 7 a 12: fornecem informações sobre o sujeito da asserção, para o qual se aplica todas as declarações contidas. O sujeito tem um identificador de nome (linha 10) cujo valor é “j.doe@example.com”,

fornecido no formato descrito na linha 9 (endereço de email). A SAML define vários formatos para identificadores de nome e ainda permite novas definições.

A asserção de um modo geral possui um período de validade indicado pelas linhas 14 e 15. Condições adicionais para o uso da asserção podem ser fornecidas dentro deste elemento, a SAML predefine algumas e também podem ser definidas outras. Os *timestamps* na SAML usam o tipo de dado `dateTime` do XML *Schema*.

A declaração de autenticação que aparece nas linhas 17 a 24 mostra que este sujeito foi autenticado originalmente usando um mecanismo de transporte de senha protegida (e.g., passando um nome de usuário e senha submetidos sobre uma sessão do navegador protegida por SSL) com o horário e data indicados. A SAML predefine vários mecanismos de contexto de autenticação (chamados classes), mas podem ser criados outros.

O elemento `<NameID>` dentro de um `<Subject>` oferece a capacidade de fornecer identificadores de nomes em diferentes formatos. Alguns dos formatos predefinidos são: Endereço de email; Nome do sujeito X.509; Nome qualificado do domínio do Windows; Nome do principal do Kerberos; Identificador permanente; Identificador temporário.

Destes, os identificadores de nome permanentes e temporários utilizam pseudônimos de preservação de privacidade para representar o principal. Os identificadores permanentes fornecem uma federação de preservação de privacidade permanente, uma vez que se mantém ligados com as identidades locais até que sejam explicitamente removidos. Os identificadores temporários suportam “anonimato” em um SP, uma vez que correspondem a um identificador de “uso único” criado no IdP. Eles não são associados com uma identidade local específica de usuário no SP e são destruídos quando a sessão do usuário termina.

Quando identificadores permanentes são criados por um IdP, normalmente são para o uso de um único SP. Isto é, um SP só irá saber sobre o identificador permanente criado pelo IdP para um principal quando consultar esse SP. O SP não conhece os identificadores para um mesmo principal que o IdP pode ter criado para o usuário em outros provedores de serviço. A SAML, no entanto, também oferece suporte para o conceito de associação (*affiliation*) de provedores de serviços que podem compartilhar um identificador permanente único para identificar um principal. Isto proporciona um meio de um SP utilizar diretamente serviços de outro SP na associação em nome do principal. Sem uma

associação, os provedores de serviços irão depender do protocolo de Mapeamento de Identificador de Nome e sempre interagir com o IdP para obter um identificador que possa ser usado em algum outro SP específico.

Uma asserção contém algumas informações básicas obrigatórias e opcionais que se aplica a todas as asserções, e geralmente um sujeito da asserção, condições usadas para validar a asserção e declarações de asserção. Apesar do exemplo anterior conter apenas uma declaração de autenticação, asserções podem conter diversas declarações da mesma autoridade sobre o mesmo sujeito. Os tipos de declaração definidos são os seguintes:

Declarações de autenticação (*Authentication statements*): são criadas pela parte que autenticou um usuário com êxito. No mínimo, elas descrevem os meios específicos usados para autenticar o usuário (usuário/senha, assinatura digital, etc.) e o momento específico que a autenticação ocorreu.

Declarações de atributos (*Attribute statements*): contém atributos de identificação específicos sobre o sujeito como nome, filiação, certificado digital, etc. A SAML permite a utilização de gramáticas de atributos quaisquer, no entanto, provê perfis para a utilização de esquemas importantes como UUID (LEACH, MEALLING e SALZ, 2005), X.500/LDAP (ITU-T, 2001)(HODGES e MORGAN, 2002) e XACML (OASIS, 2005);

Declarações de decisão de autorização (*Authorization decision statements*): definem algo que o sujeito tem direito de fazer sobre determinados recursos.

Protocolos

Os protocolos estão relacionados com a troca de asserções SAML, gerenciamento de contextos de segurança (sessões de autenticação) e gerenciamento de identificadores de usuários. Os protocolos SAML são definidos em duas camadas, sendo a camada superior formada pelos esquemas XML das mensagens e a camada inferior composta de especificações de como usar protocolos subjacentes (como SOAP e HTTP, por exemplo) para transportar essas mensagens. Isso deve garantir a interoperabilidade das aplicações e a possibilidade de utilizar os protocolos SAML em diversos cenários distintos nos quais as aplicações possuem restrições específicas. Por exemplo, um navegador de páginas da web fornece suporte a protocolos

de transporte diferentes de um cliente SOAP. Os protocolos definidos pelas especificações são descritos a seguir:

Protocolos da camada superior (esquemas XML)

Protocolo de Pedido de Autenticação (*Authentication Request Protocol*): define um meio pelo qual um principal (ou um agente atuando em nome do principal) pode requisitar asserções contendo declarações de autenticação e, opcionalmente, declarações de atributos. O Perfil SSO de navegador da web usa este protocolo ao redirecionar o usuário a partir de um SP para um IdP, quando precisa obter uma asserção para estabelecer um contexto de segurança no SP para o usuário.

Protocolo de Encerramento Único de Sessão (*Single Logout Protocol*): define um mecanismo que possibilite o encerramento quase simultâneo das sessões ativas associadas com o principal (e os recursos associados a esta sejam liberados). O encerramento pode ser iniciado diretamente pelo usuário, ou iniciado por um IdP ou SP por causa de um tempo limite da sessão, ou por um comando do administrador, etc.

Protocolo de Consulta e Pedido de Asserção (*Assertion Query and Request Protocol*): define um conjunto de consultas através das quais as asserções SAML podem ser obtidas. O formulário Pedido (*Request form*) deste protocolo pode pedir a uma parte declarante (*asserting party*) por uma asserção existente referindo-se à ID da asserção. O formulário `Consulta` (*Query form*) deste protocolo define como uma parte confiante pode solicitar asserções (novas ou existentes) com base em um sujeito específico e o tipo de declaração desejado.

Protocolo de Resolução de Artefatos (*Artifact Resolution Protocol*): fornece um mecanismo pelo qual as mensagens do protocolo SAML podem ser passadas por referência usando um identificador de tamanho fixo chamado artefato. De posse do artefato, é possível invocar o emissor da mensagem para obter o conteúdo da mensagem.

Protocolo de Gerenciamento de Identificador de Nome (*Name Identifier Management Protocol*): fornece mecanismos para mudar o valor ou o formato do identificador de nome usado para se referir ao principal. O emissor do pedido pode ser ou o provedor de serviços ou o provedor de identidades. O protocolo também provê um mecanismo de terminar uma associação de um identificador de nome entre um provedor de identidades e um provedor de serviços.

Protocolo para Mapeamento de Identificador de Nome (*Name Identifier Mapping Protocol*): fornece um mecanismo de mapear

programaticamente um identificador de nome SAML em outro, submetido a controles de política apropriados. Ele permite, por exemplo, um SP requerer de um IdP um identificador para um usuário que o SP pode usar em outro SP em um cenário de integração de aplicações. Ou seja, relaciona identificadores diferentes associados a um mesmo sujeito por aplicações diferentes.

Protocolos da camada inferior (mapeamentos para protocolos de transporte)

Redirecionamento HTTP (*HTTP Redirect Binding*): define como mensagens SAML podem ser transportadas usando mensagens de redirecionamento HTTP.

HTTP POST (*HTTP POST Binding*): define como mensagens SAML podem ser transportadas codificadas em base 64 dentro de formulários HTML.

Artefato HTTP (*HTTP Artifact Binding*): define como um artefato é transportado de um remetente para um destinatário usando HTTP através de formulário HTML ou codificado na própria URL.

SAML sobre SOAP (*SAML SOAP Binding*): define como mensagens dos protocolos SAML são transportadas em envelopes SOAP sobre HTTP.

SOAP reverso (*Reverse SOAP (PAOS) Binding*): define uma troca de mensagens SOAP/HTTP multi estágio que permite a um cliente HTTP ser uma resposta SOAP. Usado no Perfil Cliente e Proxy Avançados e particularmente projetados para suportar gateways WAP.

SAML URI (*SAML URI Binding*): define um meio para recuperar uma asserção SAML existente, a partir de um URI (*Uniform Resource Identifier*).

Perfis

Os perfis SAML definem como asserções, protocolos e *transportes* SAML são combinados para prover melhor interoperabilidade em cenários de uso específico. São eles:

Perfil de SSO da Web (*Web Browser SSO Profile*): define como entidades SAML usam o Protocolo de Pedido de Autenticação e mensagens e asserções de resposta SAML para realizar *single sign-on* em navegadores web comuns. Este perfil define como as mensagens são usadas em combinação com os *transportes* Redirecionamento HTTP, HTTP POST, e Artefato HTTP.

Perfil de Cliente e Proxy Avançados (*Enhanced Client and Proxy (ECP) Profile*): define um perfil SSO especializado onde clientes especializados ou *proxies gateways* podem usar os *transportes* SOAP e SOAP reverso (PAOS).

Perfil de Descoberta de Provedor de Identidades (*Identity Provider Discovery Profile*): define um mecanismo possível para que um provedor de serviço fique sabendo sobre os provedores de identidades que um usuário visitou anteriormente.

Perfil de Encerramento Único de Sessão (*Single Logout Profile*): define como o Protocolo de Encerramento Único de Sessão pode ser usado com *transportes* SOAP, Redirecionamento HTTP, HTTP POST e Artefato HTTP.

Perfil de Consulta/Solicitação de Asserção (*Assertion Query/Request Profile*): define como entidades SAML podem usar o Protocolo SAML de Consulta e Solicitação para obter asserções SAML sobre um *binding* síncrono, como SOAP.

Perfil de Resolução de Artefato (*Artifact Resolution Profile*): define como entidades SAML podem usar o Protocolo de Resolução de Artefatos sobre um *transporte* síncrono, como SOAP, para obter a mensagem do protocolo a que o artefato se refere.

Perfil de Gerenciamento de Identificador de nome (*Name Identifier Management Profile*): define como o Protocolo de Gerenciamento de Identificador de Nome pode ser usado com *transportes* SOAP, Redirecionamento HTTP, HTTP POST e Artefato HTTP.

Perfil de Mapeamento de Identificador de Nome (*Name Identifier Mapping Profile*): define como o Protocolo de Mapeamento de Identificador de Nome usa um *transporte* síncrono como SOAP.

Metadados

Os cenários, em relação aos perfis, podem conter muitas entidades assumindo diferentes papéis. Essas entidades devem entrar em acordo quanto a diversos parâmetros como os identificadores das entidades (URI), os protocolos de transporte suportados, os certificados e chaves criptográficas, entre outros. Para garantir que essas informações sejam descritas e obtidas de maneira padronizada, é definido um formato em XML para expressar os metadados das entidades, bem como os perfis para a troca dinâmica dessas informações entre as entidades do sistema.

Os papéis, que as entidades do sistema podem assumir, são: provedor de identidade (*identity provider* - IdP), uma autoridade SAML responsável por autenticar sujeitos e atestar seus atributos de identidade; provedor de serviço (*service provider* - SP), uma aplicação que consome asserções SAML emitidas por um IdP e, com base nas informações contidas nestas, controla o acesso do sujeito aos recursos; autoridade de autenticação (*authentication authority*), uma autoridade SAML que implementa o protocolo de consulta de asserções contendo declarações de autenticação; ponto de decisão de política (*policy decision point* - PDP), uma autoridade SAML que implementa o protocolo de consulta de asserções contendo declarações de autorização; autoridade de atributos (*attribute authority* - AA), uma autoridade SAML que implementa o protocolo de consulta de asserções contendo declarações de atributos.

3.3 SHIBBOLETH

O *Shibboleth* é um software *middleware* que tem como objetivo criar uma estrutura segura para simplificar o gerenciamento de identidades dos usuários. O projeto *Shibboleth* (SCAVO e CANTOR, 2005) foi uma iniciativa do consórcio americano Internet2 que teve como principal objetivo lançar uma implementação de código aberto, baseada em padrões abertos, para tratar desafios relacionados ao gerenciamento de identidades e controle de acesso em instituições acadêmicas. Em 2003, foi liberada a versão 1.0 e, atualmente, o projeto *Shibboleth* está em sua segunda versão, liberada em 2008. O atual desenvolvimento do *Shibboleth* visa que este seja uma solução genérica para o gerenciamento de identidades federadas, podendo ser adotada por qualquer tipo de organização.

O *Shibboleth* é um projeto da Internet2 e do MACE (*Middleware Architecture Committee for Education*), uma união de 200 universidades americanas e européias, em conjunto com indústrias e governo para desenvolver e distribuir aplicações avançadas de rede.

O conceito de identidade federada é uma característica que permite às instituições federadas utilizarem métodos distintos e próprios de autenticação e autorização. O *Shibboleth* está fundamentado sobre padrões abertos como a XML e a SAML (*Security Assertion Markup Language*) e possibilita uma maneira simples para que aplicações web usufruam das facilidades providas pelo modelo de identidades

federadas, como o conceito de autenticação única (*Single Sign-On* – SSO) e a troca segura de atributos de usuários em todos os provedores de serviço que compõem a federação. Além de beneficiar a instituição, o conceito de identidade federada pode ajudar os usuários, pois reduz o número de senhas que os mesmos precisam lembrar. Usando o *Shibboleth* é simplificada a gerência da identidade, tanto para o usuário quanto para os provedores de serviço.

O *Shibboleth* tem como ênfase a privacidade dos atributos dos usuários, sendo que a liberação desses atributos para os provedores de serviços está condicionada às políticas de privacidade da instituição de origem do usuário e também às preferências pessoais deste usuário. O *Shibboleth* permite que os usuários controlem quais as informações serão liberadas e quem poderá recebê-las e disponibiliza um caminho para referenciar o usuário sem revelar a identidade principal. O usuário é conhecido pelo site provedor de serviço somente por um *handle* temporário, sendo assim este site pode decidir o que é permitido para o usuário.

Dentro de um domínio *Shibboleth* são encontrados dois componentes principais: provedor de identidades (*Identity Provider*, IdP) e provedor de serviços (*Service Provider*, SP). O IdP é responsável pela autenticação de usuários, o que possibilita posteriormente o uso dos serviços oferecidos pelo provedor de serviços. Ambos os componentes devem implementar toda a pilha de software fornecida pelo projeto *Shibboleth*, permitindo assim o transporte das credenciais dos usuários do provedor de identidades até o provedor de serviços (SCAVO e CANTOR, 2005). No *Shibboleth*, o processo de autenticação sempre é executado na instituição de origem do usuário, através de seu provedor de identidades, fazendo uso dos mecanismos de autenticação presentes nessa instituição. A autenticação de usuários pode ser feita através de nome de usuário e senha, *Kerberos*, X.509, ou outro (CHADWICK, 2009).

Quando um usuário tenta acessar um recurso web protegido por *Shibboleth*, ele é redirecionado para o serviço WAYF (*Where Are You From*) que questiona em qual instituição deverá ser feita a autenticação. Depois de autenticar o usuário, o IdP (*Identity Provider* – Provedor de Identidade), irá gerar uma referência temporária para identificar a autenticação do usuário e irá enviá-la para o Provedor de Serviço (SP – *Service Provider*).

O provedor de identidade por realizar a autenticação, armazena os atributos dos usuários e faz o controle destes atributos. Existem quatro componentes principais no provedor de identidade: o HS (*Handle*

Service), o AA (*Attribute Authority*), o serviço de diretório e o mecanismo de autenticação. O HS realiza a autenticação dos usuários em conjunto com o mecanismo de autenticação e cria um *handle*. O AA verifica as políticas de privacidade sobre a liberação dos atributos e permite que o usuário especifique quais provedores de serviço podem acessá-los. Já o serviço de diretório é o local onde ficam armazenados os atributos dos usuários.

O provedor de serviço é o local onde são armazenados os recursos web acessados pelos usuários, validando os *handles* enviados pelo provedor de identidade e usando-os para criar um contexto seguro. Há três componentes principais no provedor de serviço: o ACS (*Assertion Consumer Service*), o AR (*Attribute Requester*) e o RM (*Resource Manager* – Gerenciador de Recurso). O componente ACS fica responsável por receber os *handles* e em seguida estabelecer um contexto seguro. O AR é responsável por obter os atributos dos usuários, verificar as políticas de privacidade sobre a aceitação dos atributos e repassar os atributos autorizados para o RM. Finalmente, o RM é um componente que toma as decisões de controle de acesso.

A Figura 12 apresenta o fluxo durante uma operação completa do *Shibboleth*, partindo da situação em que o navegador do usuário chega ao site provedor de serviço sem uma sessão existente e sem nenhuma informação sobre seu site provedor de identidade. Os atores principais incluem: o usuário, ou seja, aquele que deseja usar o recurso web protegido; o site provedor de serviço, local onde está instalado o software SP; e o site provedor de identidade, local onde está instalado o software IdP.

O usuário navega para o recurso web usando o navegador. O recurso web, localizado no site provedor de serviço, é protegido, então se torna necessário coletar algumas informações sobre o usuário para decidir se o acesso é permitido (Passo 1). O software *Shibboleth* redireciona o navegador do usuário para uma página de navegação, chamada WAYF, nesta página é apresentada ao usuário uma lista de organizações que podem acessar o recurso web protegido (Passos 2 e 3). A partir desta etapa, o usuário seleciona a sua organização (Passo 4) e o navegador é enviado para o web site da organização escolhida, ou seja, para o componente HS localizado no seu provedor de identidade (Passo 5). O usuário visualiza a interface web de autenticação da sua organização (Passo 6), onde serão inseridas as credenciais (Passo 7). Após enviar suas credencias, o componente HS usa o mecanismo de autenticação local da organização para verificar as credenciais do usuário (Passo 8). O HS cria um *handle* para identificar o usuário. Este

handle é registrado com o AA de forma que possa ser mapeado para os atributos do usuário quando a requisição do recurso chegar. O *handle* é colocado dentro de um *handle token* e enviado para o recurso web. Este *handle token* informa que o usuário está autenticado (Passo 9). O *handle token* é recebido pelo componente ACS que examina o *handle token* para determinar o provedor de identidade que pode fornecer os atributos sobre o usuário. Após examinar e validar o *handle token*, o ACS transfere-o para o AR e cria uma sessão (Passo 10). O AR utiliza o *handle token* para requisitar ao provedor de identidade, especificamente ao AA, informações adicionais (atributos) sobre o usuário (Passo 11). No provedor de identidade, após terem sido comparados e validados o *handle token* e o *handle*, foi mapeado a identidade do usuário, sendo assim o ARP é consultado. Se a liberação dos atributos do usuário for permitida, os valores dos atributos requisitados são devolvidos (Passo 12). O AA responde com os valores dos atributos solicitados no formato SAML *assertion* (Passo 13). O provedor de serviço recebe os atributos e passa para o RM (Passo 14). O RM utiliza estes atributos para decidir se garante ou não o acesso ao recurso. Se permitido, carrega o recurso solicitado no browser do usuário (Passo 15).

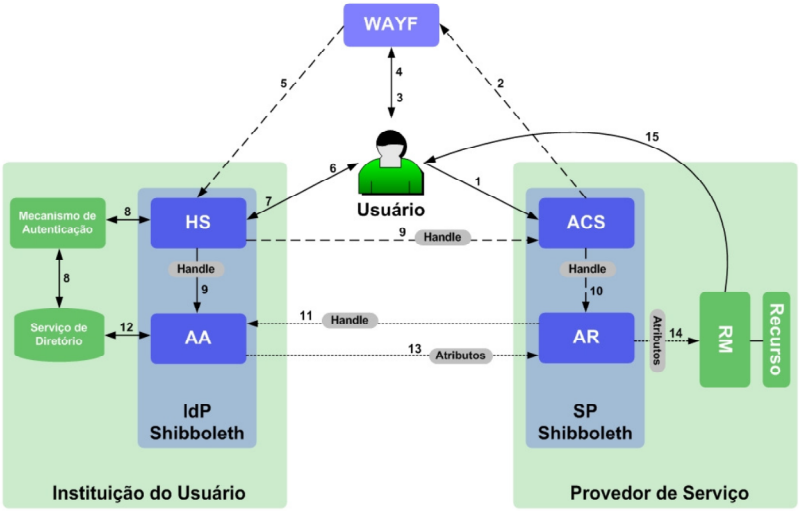

Figura 12. Diagrama de Fluxo do Sistema Shibboleth (CORDOVA e WESTPHALL, 2006)

4 SISTEMA DE AUTORIZAÇÃO NA NUVEM USANDO SHIBBOLETH

Em ambientes de computação abertos, os recursos são distribuídos em organizações distintas. Na maioria das vezes os usuários são externos. Para garantir a segurança dos recursos, os serviços devem identificar as identidades de usuários e autorizar adequadamente o acesso de acordo com a informação da identidade do usuário. Assim, o usuário deve submeter suas informações de identidade, sensíveis ou não, toda vez que acessar um novo recurso. Se estas informações forem transferidas com frequência aumentará a possibilidade de perda de privacidade e o risco de roubo de identidade. O gerenciamento de identidades pode minimizar tais problemas.

O modelo de serviço operacional e as tecnologias utilizadas para prover os serviços do ambiente de computação em nuvem apresentam diferentes níveis de riscos se comparado ao ambiente tradicional de tecnologia de informação (CSA, 2009). O provimento de recursos sob demanda para o processamento e armazenamento massivo de dados está sujeito a falhas de segurança, abusos com relação à privacidade e violação de direitos autorais. Preocupações com estes aspectos de segurança computacional estão impedindo a ampla adoção da computação em nuvem.

Para que as organizações consumidoras utilizem os serviços oferecidos pela nuvem é necessária a implantação de um modelo de gerenciamento seguro e confiável. O esquema a ser utilizado deve facilitar a inserção e remoção de usuários dos serviços oferecidos pela nuvem. A implantação de mecanismos de autenticação robustos e esquemas de delegação de direitos funcionando de maneira confiável são fundamentais para o correto gerenciamento de identidades e para a prestação de serviços em nuvens computacionais.

4.1 GERENCIAMENTO DE IDENTIDADES NA COMPUTAÇÃO EM NUVEM

Segundo (BERTINO e TAKAHASHI, 2011) para garantir o controle de acesso em ambientes abertos, como as nuvens computacionais, o gerenciamento de identidades pode ser implementado com diferentes configurações. Por exemplo, o gerenciamento de

identidades pode ser implementado internamente (*in-house*), conforme a Figura 13(a). Nesta configuração, as identidades são emitidas e gerenciadas pelas empresas clientes dos serviços em nuvem. Outra maneira é terceirizar o serviço de gerenciamento de identidades, que é mais conhecido como Identidade como Serviço (*Identity as a Service*, IDaaS), conforme a Figura 13(b). Nesta configuração as identidades são emitidas e gerenciadas por um terceiro confiável. Por último tem-se o caso em que cada provedor de serviço de nuvem implementa suas próprias funções de gerenciamento de identidades independentemente, conforme a Figura 13(c).

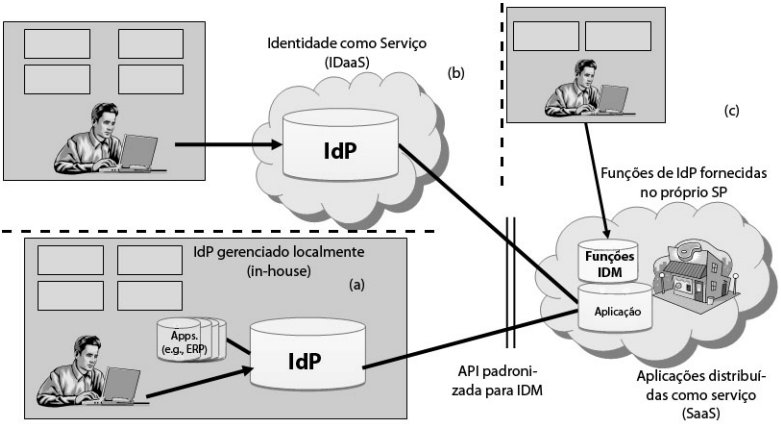

Figura 13. Configurações de sistemas de IDM em ambientes de computação em nuvem (BERTINO e TAKAHASHI, 2011).

Nesta dissertação optou-se por usar o primeiro caso de configuração (Figura 13(a)), onde a empresa cliente tem total domínio e responsabilidade sobre as identidades digitais de seus usuários. Para apoiar a ideia, (CSA, 2010a) e (MARCONJR, LAUREANO, *et al.*, 2010) também citam a possibilidade de usar o gerenciamento de identidades interno, em contraposição às outras configurações possíveis (Figuras 13(b) e 13(c)).

1. "Identidade como um serviço em nuvem (*Cloud Identity as a Service* – IDaaS) é o gerenciamento de identidades na nuvem, sendo externo as aplicações e aos provedores que utilizam as identidades. O serviço é fornecido por terceiros com funções de gerenciamento de identidades e controle de acesso, incluindo gerenciamento do ciclo de vida de identidades e acesso único (*Single Sign-On* – SSO). O termo

é bastante amplo e engloba serviços de software, plataforma ou infraestrutura, tanto para nuvens públicas como privadas. Soluções híbridas também são possíveis, segundo as quais, identidades podem ser gerenciadas internamente dentro de uma organização, enquanto outros componentes, como autenticação, são terceirizados através de uma Arquitetura Orientada a Serviços (*Service Oriented Architecture* – SOA)."

2. "Provedores de serviço nos níveis SaaS e PaaS geralmente oferecem serviços de autenticação acoplados as suas aplicações e plataformas. Alternativamente os provedores podem delegar a autenticação dos usuários para a organização contratante dos serviços. Com esta abordagem, o contratante pode autenticar seus usuários localmente, utilizando um serviço de identificação interno à organização e estabelecendo confiança com o fornecedor de serviço através da federação de identidades, por exemplo".

A seguir são apresentadas as descrições de soluções propostas em outros dois trabalhos, os quais serviram de base para a realização da nova proposta. A primeira descreve sobre os locais das políticas de autorização e a segunda descreve sobre as formas de autorização no ambiente.

4.1.1 Locais das políticas de autorização

A descrição a seguir refere-se ao trabalho de (MARCONJR, LAUREANO, *et al.*, 2010) e o principal destaque é a criação de políticas e armazenamento das mesmas no ambiente cliente e sua posterior utilização pelo CSP.

Os serviços fornecidos pela nuvem computacional podem ser disponibilizados em qualquer local físico de abrangência da mesma. A gerência de um grande número de serviços (SaaS, PaaS, IaaS) e recursos físicos pode gerar um volume considerável de dados a ser administrada de maneira centralizada, pois será necessário coletar, armazenar, analisar e processar estes dados. Assim, a administração centralizada pode ser considerada impraticável, e portanto faz-se necessário instanciar serviços de gerenciamento distribuídos e fracamente acoplados (com baixa dependência funcional).

O ambiente de computação em nuvem está sendo utilizado para hospedar vários tipos de serviços, e todos exigem garantias de segurança dos dados sendo processados e armazenados. Para tirar o máximo proveito de todo o poder oferecido pela nuvem computacional, as diferentes entidades – provedores e consumidores de serviços – que interagem com o ambiente necessitam de abordagens de segurança abrangentes e confiáveis. O particionamento do ambiente de computação em nuvem em diferentes domínios cria escopos de proteção reduzidos, regrando e limitando as interações entre as partes, classificando os tipos de serviços e recursos, facilitando as operações de gerenciamento e efetuando o balanceamento e a distribuição de carga.

Para o ambiente de nuvem o gerenciamento da autenticação deve fornecer suporte aos processos de criação e emissão das credenciais (e.g. senhas, certificados digitais, credenciais dinâmicas) utilizadas pelos usuários da organização. Procedimentos de autenticação robustos (e.g. baseada em múltiplos fatores) podem não ser compatíveis com determinados serviços fornecidos pela nuvem. Consequentemente, a utilização de uma grande variedade de métodos de autenticação gerará carga administrativa adicional. O usuário dos serviços também precisa ter a usabilidade considerada, pois esse pode necessitar utilizar um conjunto de métodos para as aplicações internas a organização, e outro conjunto para acessar os serviços na nuvem. O mesmo desafio aplica-se aos provedores de computação em nuvem, pois o custo para suportar vários mecanismos de autenticação, acomodando as necessidades de consumidores usando mecanismos heterogêneos pode se tornar pouco atrativo para a entidade que mantém a nuvem. Neste caso, o ideal é a padronização dos mecanismos de autenticação para resolver estas limitações impostas pelas características de computação em nuvem.

A definição de um sistema de segurança baseado em políticas é uma necessidade administrativa e de uso da nuvem computacional, pois é possível controlar o acesso e uso individual de cada usuário do ambiente. Cada organização consumidora de serviços fornecidos pela nuvem precisa definir políticas para seus usuários. Os seguintes tipos de usuários devem ser considerados nas políticas: administrador, desenvolvedor e usuários finais da organização. Adicionalmente, práticas, processos e procedimentos de gerenciamento de identidade e acesso devem englobar os serviços oferecidos pela nuvem. O gerenciamento deve ser, preferencialmente, escalável, efetivo e eficiente para ambos, provedores e consumidores dos serviços.

A utilização de serviços dentro da nuvem cria a possibilidade das políticas de controle de acesso serem definidas em um lugar – por exemplo, internamente a organização (Figura 14; evento *poc*; Repositório de Política do Consumidor – RPC) – e serem executadas em outro (evento *pop*; Repositório de Política do Provedor – RPP). Isto é, as políticas definidas pela organização consumidora podem ser transferidas do repositório de políticas para o provedor de computação em nuvem que controla o uso dos serviços. Padrões de serviços web como a XACML – *eXtensible Access Control Markup Language* e a *WS-Policy* são úteis nestes casos. Porém, mesmo com a utilização de especificações padronizadas, o consumidor e o provedor de computação em nuvem precisam utilizar a mesma semântica para que a interação entre as entidades dos diferentes domínios possa ocorrer de maneira transparente e segura – e.g. solicitações de acesso aos serviços (evento *ac*).

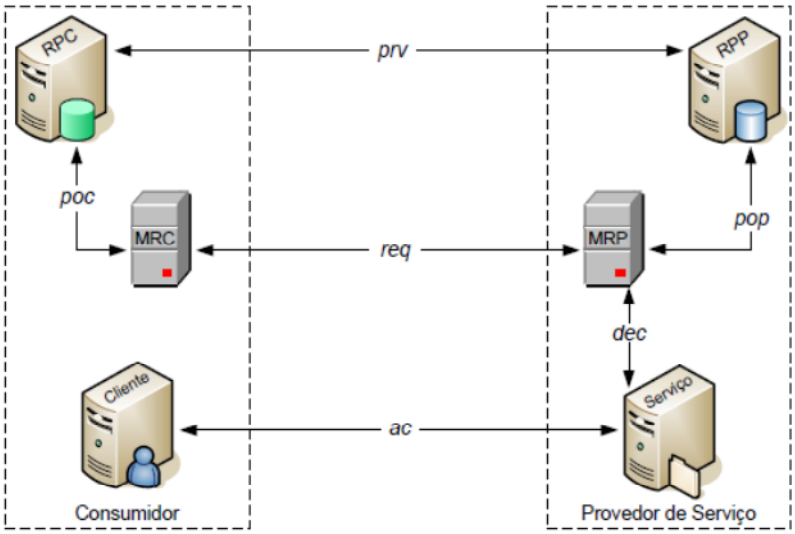

Figura 14. Provisionamento de políticas no provedor **(MARCONJR, LAUREANO, *et al.*, 2010)**.

A transferência ou configuração de políticas pode ocorrer periodicamente, no modo batch ou sob demanda (*just-in-time*), quando será enviada concomitantemente com a solicitação de configuração vinda do Monitor de Referência do Provedor (MRP; evento *prv*). Para o modo batch a especificação SPML (*Service Provisioning Markup Language*) pode ser utilizada. Porém, se o modo *single sign-on* estiver ativo e o provedor de serviço de computação em nuvem for capaz de

receber informações de políticas em assertivas SAML, sob demanda, então as informações podem ser transmitidas utilizando o *SAML profile of XACML*. Adicionalmente, se as decisões de autorização forem avaliadas externamente aos serviços hospedados na nuvem (i.e. no Monitor de Referência do Consumidor – MRC), o padrão XACML pode ser utilizado para expressar solicitações e respostas de avaliações de políticas (evento *req*).

O provisionamento (pré-configuração) de políticas para consumidores pode ser efetuado com a utilização da especificação SPML. As políticas podem conter papeis de acesso pré-definidos: administrador, desenvolvedor e usuário final.

Os consumidores precisam ter certeza que os provedores de nuvem suportam suas necessidades e fornecem mecanismos de controle de acordo com a dinâmica exigida pelo ambiente. Basicamente, o provedor de computação em nuvem precisa: controlar o acesso de usuários aos serviços fornecidos pela nuvem – de acordo com as políticas definidas pelo consumidor; honrar os SLAs ou contratos de QoS estabelecidos com organização consumidora; controlar o acesso aos dados dos usuários; controlar acesso a área do sistema no nível de usuário e administrativo (privilegiado); manter as informações do perfil do usuário e as políticas de controle de acesso atualizadas; permitir a coleta de informações do perfil do usuário e das políticas de controle de acesso implantadas no provedor (para um determinado consumidor); fornecer meios de notificação para alterações em contas de usuários (e.g. criação, remoção, concessões de acesso), visando coibir a configuração de contas falsas ou modificação de direitos de acesso no provedor sem que o consumidor saiba; e fornecer trilhas de auditoria para o ambiente de cada consumidor – identificando atividades de gerenciamento e acesso, assim como a utilização de qualquer recurso para qual foram estabelecidas cotas de uso.

4.1.2 Formas de autorização do ambiente

A descrição a seguir refere-se ao trabalho de (CALERO, EDWARDS, *et al.*, 2010) e o principal destaque é a elaboração e posteriormente uma implementação de um sistema de autorização *multi-tenancy* que fornece controle de acesso às informações e serviços de todos os diferentes serviços em nuvem, usando a infraestrutura de nuvem.

O trabalho referido descreve um sistema de autorização *multi-tenancy* adequado para serviços de *middleware* na camada de PaaS. Esse sistema de autorização fornece controle de acesso às informações e serviços de todos os diferentes serviços em nuvem, usando a infraestrutura de nuvem. Cada empresa pode fornecer vários serviços em nuvem que podem colaborar com outros serviços que pertencem ou à mesma organização ou a uma organização diferente. Esse sistema de autorização é capaz de suportar estes acordos de cooperação entre empresas (também conhecido como uma federação).

A criação de um modelo de autorização, para controlar o acesso a recursos em um sistema de nuvem, deve determinar se um sujeito tem o privilégio de realizar uma determinada ação sobre o objeto controlado. Isto pode ser representado usando uma 3-tupla `(Subject, Privilege, Object)`. No entanto, essa 3-tupla deve ser estendida para habilitar o suporte a *multi-tenancy*, onde diferentes emissores estão usando o sistema de autorização simultaneamente. Assim, pode-se definir uma 4-tupla `(Issuer, Subject, Privilege, Object)`. Além disso, o modelo de autorização pode ser estendido para proteger diferentes tipos de objetos, aumentando a 4-tupla anterior em uma 5-tupla `(Issuer, Subject, Privilege, Interface, Object)`, onde *Interface* representa a forma como um objeto é interpretado. Então, o sistema pode interpretar essa 5-tupla da seguinte forma: o `Emissor` (Issuer) diz que o `Sujeito` (Subject) tem o `Privilégio` (Privilege) de realizar uma determinada ação sobre o `Objeto` (Object) associado ao tipo de `Interface` (Interface). Por exemplo, o sistema pode interpretar `(Jose, Nigel, Read, CloudStorage, \root\)` como `José` diz que `Nigel` pode ler a pasta `\root\` associada ao serviço `CloudStorage`. É importante observar que um privilégio pode permitir múltiplas ações, por exemplo, escrever pode permitir as ações de exclusão e atualização.

Para apoiar o modelo de autorização, foi incorporado o controle de acesso baseado em papéis (RBAC). O modelo de autorização depende da utilização de papéis como um conjunto de privilégios que poderiam ser atribuídos a um usuário. Assim, pode-se definir a adesão de um usuário para uma determinada função, por meio de uma 3-tupla `(Issuer, User, roleName)`, que o sistema pode interpretar como o `Emissor` diz que o `Usuário` tem o `Papel` dado. Um papel é delimitado pelo emissor, e então é representado na forma `papel(Issuer, roleName)`, para representar um papel. Daí, o

papel(Jose, Admin) e o papel(Nigel, Admin) serem diferentes. Na primeira, José está emitindo o Papel Admin, e no segundo, Nigel está emitindo o Papel Admin. Um serviço de autorização pode tratar estes dois papéis de forma diferente. Quaisquer privilégios concedidos a um papel são concedidos aos usuários com esse papel.

Uma relação transitiva existe entre o papel e o usuário, de modo que o sistema de autorização deve inferir a relação correta. Por exemplo, o sistema pode interpretar (Jose, Nigel, DatabaseAdmin) como José diz que o usuário Nigel pertence à função (papel) DatabaseAdmin. Para maior clareza, devemos estender a 5-tupla anterior para definir privilégios sobre os papéis, assim se define o Sujeito (Subject) como Usuário (User) ou Papel (Role). O resultado é uma nova 5-tupla definida como (Issuer, [User|Role], Privilege, Interface, Object). Para evitar qualquer ambiguidade, representa-se o usuário e o papel como um tipo(valor) nesta 5-tupla, por exemplo, pode-se dizer User(Nigel) e Role(Jose, DatabaseAdmin), respectivamente. Então, o sistema pode interpretar (José, role(Nigel, Admin), Read, CloudStorage, \root\) como José diz que o papel(Nigel, Admin) pode ler a pasta \root\ do CloudStorage.

Além disso, é incorporado o controle de acesso baseado em papel hierárquico (hRBAC) o que permite um agrupamento hierárquico de sujeitos, dentro do modelo de autorização. Este tipo de controle de acesso permite a definição de hierarquias de papéis. Um papel pode ser definido como uma especialização de um outro mais genérico, por exemplo, pode-se definir o papel(Jose, DatabaseAdmin) como uma especialização de papel(Nigel, Admin), e, portanto, todos os privilégios definidos sobre o papel(Nigel, Admin) também são concedidos para o papel(Jose, DatabaseAdmin). Para este fim, é estendida a 3-tupla anterior como (Issuer, [User|Role], Role), que permite definir os papéis pertencentes a outro papel. Novamente, para evitar qualquer ambiguidade, é definido o Usuário e Papel seguindo o padrão de tipo(valor). Por exemplo, o sistema pode interpretar (Nigel, role(Jose, DatabaseAdmin), Admin) como Nigel diz que papel(Jose, DatabaseAdmin) é um subpapel do papel(Nigel, Admin): todos os privilégios

concedidos ao `papel(Nigel, Admin)` também será concedido para o `papel(Jose, DatabaseAdmin)`.

Ambos os campos `User` e `Issuer` disponíveis nas 5 e 3-tupla são administrados como texto simples, que separa o modelo de autorização do método de autenticação usado para autenticar usuários e Emissores. Pode-se autenticá-los por qualquer mecanismo de autenticação, como o OpenID, X.509, ou Kerberos, entre outros.

Hierarquias de objetos fornecem poderosa expressividade no modelo de autorização. Elas podem aumentar o escopo de um privilégio aplicado a um determinado objeto a todos os objetos que são descendentes do objeto original. O uso de caminhos hierárquicos, também é uma abstração de programação comum usada em computação em nuvem, por exemplo, Transferência de Estado Representacional (*Representational State Transfer*, REST). Para suportar isto, o campo `Object` permite a representação da hierarquia de objetos por meio de caminhos, usando um símbolo especial "\". Um caminho define a estrutura hierárquica do recurso protegido, por exemplo "`\pictures\imageA`" denota que o objeto `Pictures` é um pai do objeto `imageA`. Neste caso, `Pictures` é uma pasta e `imageA` é um arquivo. No entanto, estas semânticas são fornecidas pela interface definida na 5-tupla. O símbolo "*" é usado para facilitar a definição de privilégios aplicados a hierarquias de objetos. Assim, o símbolo "*" no caminho define que o privilégio é aplicado não somente para o objeto especificado, mas também para todos os seus descendentes na hierarquia. Assim, `(José, user(Nigel), Read, CloudStorage, \root\*)` é interpretado como `José` diz que o usuário `Nigel` pode `Read` a pasta `\root\` e todas as suas subpastas no serviço `CloudStorage`. Assim, pode-se definir a 5-tupla como `(Issuer, [User|Role], Privilege, Interface, ObjectPath)`.

Para demonstrar como esse modelo suporta federação, considere um cenário de computação em nuvem com um sistema de autorização *multi-tenancy* (Figura 15). Este cenário envolve três empresas diferentes, cada uma com dois serviços em nuvem diferentes. Serviços em nuvem pertencentes a empresas distintas podem usar um modelo de informação diferente, com informações relacionadas aos usuários, privilégios, papéis e recursos. No entanto, sua federação pode exigir o compartilhamento de informações de autorização.

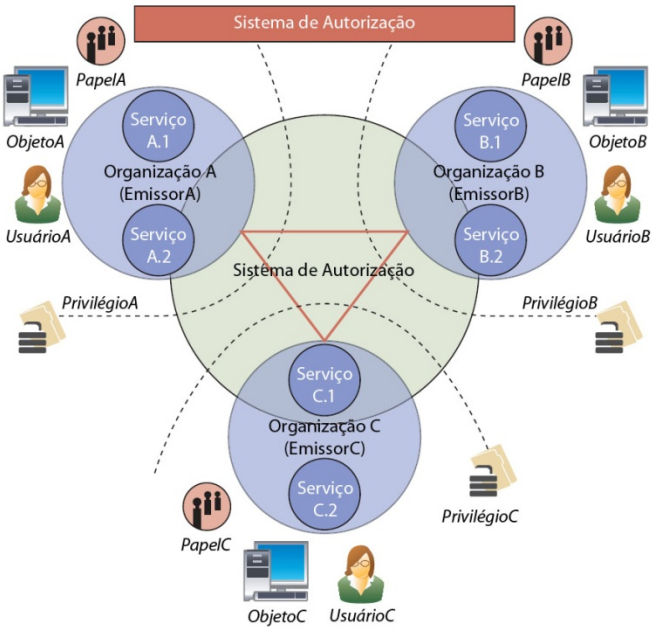

Figura 15. Exemplo de um sistema de autorização em um cenário multi-tenancy **(CALERO, EDWARDS, *et al.*, 2010)**.

Por exemplo, o sistema pode interpretar a 5-tupla `(IssuerA, role(IssuerB, users), Read, ServiceA.1, \root\)` como `IssuerA` permite todos com `papel(IssuerB, users)` o acesso de `Leitura` à pasta `\root\` do sistema de arquivos fornecido pelo `Serviço A.1`. É importante observar que `papel(IssuerB, users)` é controlado por `IssuerB`. Da mesma forma, é possível ter mais uma 5-tupla `(IssuerA, role(IssuerC, users), Read, ServiceA.1, \root\)` para "`usuários de C`". Este exemplo envolve muitas preocupações com a privacidade, porque exige que `Serviço A.1` possa acessar as declarações de autorização de ambos os `IssuerB` e `IssuerC`. Neste caso, o módulo gerenciador de confiança dá apoio para controlar questões de privacidade.

Também é possível criar todos os campos, com exceção do campo `Issuer`, com o símbolo "`*`" para se referir a todas as instâncias disponíveis no modelo de informações. Por exemplo, o sistema pode

interpretar a 3-tupla `(José, user(*), Public)` como `José` diz que todos os usuários pertencem ao `papel(Jose, Public)`.

Este modelo foi projetado para suportar *multi-tenancy*, RBAC, hRBAC, hierarquias de objetos baseadas em caminho e federação. Os diferentes emissores usam este modelo para definir as informações de autorização no sistema. Quando há um pedido de autorização, o sistema utiliza todas as informações de autorização para determinar se um pedido está autorizado. Se ele não pode comprovar a autorização usando as 5 e 3-tuplas armazenadas, então um "negar por padrão" é aplicado, rejeitando o acesso ao recurso.

4.1.3 Caso de uso da empresa para a nuvem

Em (AHRONOVITZ, AMRHEIN, *et al.*, 2010) pode ser encontrada uma descrição para os casos de uso mais comuns encontrados para o uso de nuvens. Nesta dissertação optou-se por usar o caso de uso “da empresa para a nuvem” (*Enterprise to Cloud*) na Figura 16. O elemento Usuário Final é representado através de uma linha pontilhada, pois não interage neste caso de uso.

Este caso de uso envolve uma empresa usando serviços em nuvem para seus próprios benefícios, e é muito comum, pois concede maior controle à empresa. Neste cenário a empresa usa os serviços de nuvem para complementar os recursos de que necessita: usando armazenamento em nuvem para backups ou armazenamento de dados raramente usados; usando máquinas virtuais na nuvem para aumentar o poder de processamento e lidar com cargas de pico; usando aplicações na nuvem (SaaS); usando bases de dados em nuvem como parte do processamento de um aplicativo. Isto poderia ser extremamente útil para compartilhar esse banco de dados com parceiros ou agências governamentais.

Figura 16. Cenário "da empresa para a nuvem" **(AHRONOVITZ, AMRHEIN, *et al.*, 2010)**.

4.2 MODELO DE GERENCIAMENTO DE IDENTIDADES IN-HOUSE NA COMPUTAÇÃO EM NUVEM

Após uma análise na literatura em busca de uma solução de IDM para a nuvem, optou-se por propor uma nova solução de controle de acesso, focando a autorização.

A proposta nesta dissertação é modelar um protótipo de um sistema de gerenciamento de identidade e acesso (*IAM – Identity and Access Management*) a ser usado por uma organização para acessar serviços oferecidos na nuvem. O foco desta dissertação foi voltado para uma solução alternativa à de IDaaS. No serviço IDaaS as atividades de gerenciamento de identidades são terceirizadas e, consequentemente, os dados e informações sensíveis dos usuários estão fora do domínio da

organização, sendo controladas e mantidas por esta terceira parte. A alternativa para esta solução e que foi usada neste trabalho é a chamada solução *in house*. Nesta solução as atividades de gerenciamento de identidades dispensam a necessidade de terceirização, ficando o cliente responsável por gerenciar seus próprios usuários (BERTINO e TAKAHASHI, 2011).

O sistema de gerenciamento de identidade ficará localizado na organização cliente e a comunicação com o provedor de serviços em nuvem (Cloud Service Provider, CSP) será realizada através de federação de identidades. Ao estabelecer a relação de confiança entre as partes, o provedor de serviços em nuvem irá solicitar a autenticação dos usuários ao provedor de identidades (Identity Provider, IdP) que está no cliente. Dessa forma, os dados dos usuários permanecem sob os cuidados de sua própria organização, melhorando a privacidade e evitando a perda de informações. Além disso, quando no domínio do CSP, os usuários serão conhecidos através de pseudônimos, o que deve garantir o caráter individual dos usuários sem a necessidade de expor suas informações verdadeiras.

O sistema de acesso realiza a autorização ou controle de acesso no ambiente. O CSP deve ser capaz de interpretar e liberar o acesso de acordo com os privilégios de cada usuário separadamente. A instituição terá a responsabilidade de fornecer os atributos de usuários para a aplicação implantada no provedor de serviço da nuvem. Tal como apresentado anteriormente, em “Locais das políticas de autorização”, o CSP irá receber as políticas de acesso vindas do provedor de identidades do cliente e será capaz de interpretá-las.

O sistema de autorização deverá ser capaz de aceitar múltiplos clientes, como em um modelo *multi-tenancy*. O conceito de *multi-tenancy* determina que uma aplicação seja usada igualmente por uma série de usuários, cada um recebendo níveis comparáveis ou equitativos da capacidade de resposta e largura de banda através do uso de balanceamento de carga (PALSONKENNEDY e GOPAL, 2011).

A seguir, na Figura 17, pode ser observada uma visão detalhada da estrutura proposta para ser usada como base para o controle de acesso no uso de serviços em nuvem, bem como o fluxo de comunicação entre os provedores.

O provedor de serviços que está sendo requisitado para liberar o acesso a um determinado recurso ou serviço deverá comunicar-se com o provedor de identidades correspondente. Subentende-se que uma dada requisição de acesso possui a informação de sua própria origem. Sendo

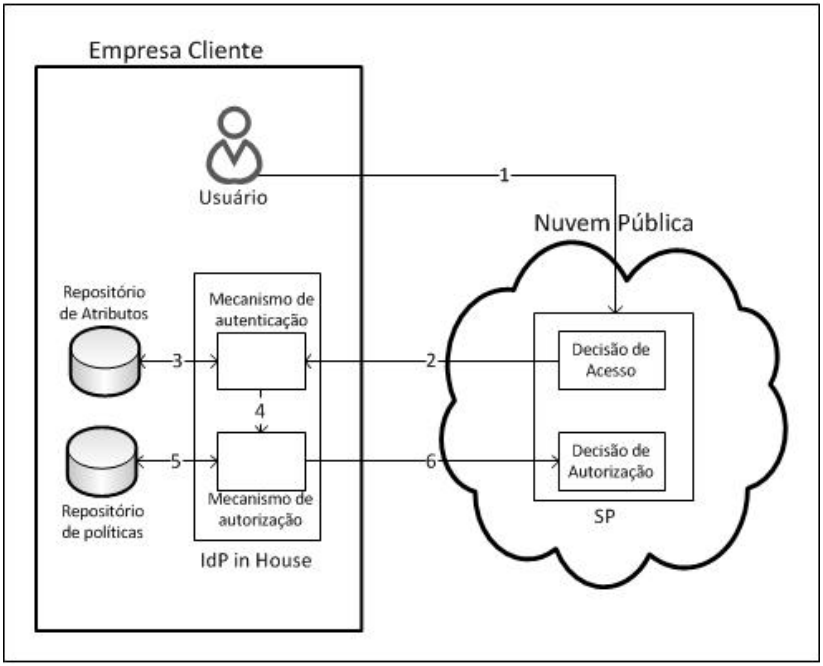

Figura 17. Estrutura detalhada do esquema de gerenciamento de identidades.

assim, o SP é capaz de interpretar esta requisição e saber qual é o IdP correspondente.

A comunicação entre os provedores (de serviço e de identidade) tem sempre a finalidade de conferir autenticidade a uma requisição para posterior liberação de acesso a um serviço ou recurso que esteja sendo solicitado.

Ao garantir a autenticidade de uma identidade o IdP fica responsável por enviar uma resposta ao SP contendo as definições de permissões para aquela identidade em questão. Isto significa enviar as políticas de acesso ao SP, onde serão, posteriormente, tomadas as decisões de autorização.

Dessa forma o SP deverá ser capaz de interpretar as políticas de acesso para fazer a autorização no ambiente. Com isso, elimina-se a necessidade de uma nova comunicação entre SP e IdP. Além disso diminui-se o overhead na comunicação e evita a exposição de informações por vezes repetidas.

A comunicação entre SP e IdP pode ser analisada na Figura 17. Primeiramente um usuário solicita ao SP o acesso a um determinado

recurso ou serviço (passo 1). O SP recebe a solicitação e verifica o tipo de acesso através do componente “Decisão de Acesso”, ou seja, se a solicitação refere-se a um acesso público ou privado. No caso de acesso público o SP irá conceder o acesso sem a necessidade de autenticação. Caso contrário o SP irá solicitar ao IdP a autenticação do usuário em questão (passo 2). O mecanismo de autenticação do IdP irá realizar a autenticação do usuário (passo 3) com base nos atributos que encontram-se em um repositório específico (repositório de atributos). Em caso de sucesso na autenticação, o próximo passo será coletar as informações de autorização referentes àquele usuário (passo 4). O mecanismo de autorização se comunica, primeiramente, com o repositório de políticas (passo 5) a fim de coletar as políticas de acesso referentes à solicitação e, posteriormente, envia uma mensagem ao SP que confirma a identidade contida na solicitação e envia junto com esta resposta as políticas de acesso para aquela identidade (passo 6). Ao receber todas as informações necessárias, o componente “Decisão de Autorização” no SP, irá realizar a autorização no ambiente.

Vale ressaltar que é no passo 6 que pretende-se usar um formato de mensagem baseado na tupla apresentada anteriormente, conforme (CALERO, EDWARDS, *et al.*, 2010). É nesta mensagem que deverão conter as políticas de acesso de uma identidade em questão. O SP deverá ser capaz de interpretar esta tupla, bem como aplicar as regras de autorização conforme as políticas de uma determinada identidade.

4.2.1 Cenário

Em relação ao cenário para implantação do sistema de controle de acesso e gerenciamento de identidades pode-se conferir a Figura 18. O cenário é o de uma federação acadêmica compartilhando recursos e/ou serviços em nuvem. Um recurso e/ou serviço é disponibilizado por uma instituição acadêmica, em um provedor de serviços de nuvem, e é compartilhado com outras instituições acadêmicas. Para poder compartilhar recursos e serviços é necessário que uma instituição esteja filiada à federação (LEANDRO, NASCIMENTO, *et al.*, 2012).

Para uma instituição se filiar à federação deverá possuir um IdP configurado de forma que atenda às exigências impostas por tal federação. Sendo estas exigências expressas na forma de políticas de acesso e privacidade definidas pela linguagem SAML.

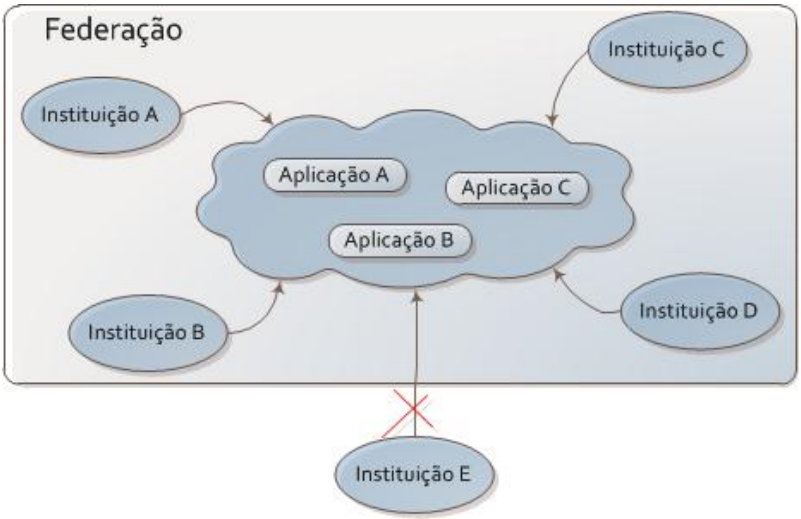

Figura 18. Federação acadêmica compartilhando serviços em nuvem.

Uma vez filiada à federação, a instituição será capaz de autenticar seus próprios usuários, conforme o funcionamento do sistema de autenticação e autorização descrito na sessão anterior. Sendo que a autorização será responsabilidade do SP.

Conforme mostra a Figura 18, uma instituição não filiada não tem permissão de acesso às aplicações de uma federação. Este é o caso da "Instituição E" representada na figura. Outro aspecto importante a ser observado é que uma instituição pode apenas estar usufruindo de serviços oferecidos na nuvem, como é o caso da Instituição D representada na figura. Esta instituição não oferece serviços, mas pode fazer parte da federação e usar os serviços oferecidos.

4.3 IMPLANTAÇÃO DO CENÁRIO PROPOSTO

A implantação do cenário proposto contou com o apoio de dois bolsistas de IC, os quais realizaram individualmente a implementação de provedores usando a infraestrutura de nuvem da Amazon. Posteriormente, ambos realizaram as alterações e configurações necessárias para que um provedor se comunicasse com o outro.

A implementação do provedor de serviços (SP) em nuvem ficou a encargo do trabalho de (NASCIMENTO e WESTPHALL, 2011). Enquanto o provedor de identidades (IdP) em nuvem foi implementado por (SANTOS e WESTPHALL, 2011). De posse dessas implementações, foi então criado um outro IdP, para representar o uso de serviços, oferecidos pelo SP, por mais de um cliente. Caracterizando dessa forma um sistema multi-tenancy.

4.3.1 Implementação do provedor de serviços

Para a realização de testes e demonstrações, primeiramente foi implantado um provedor de serviços (*Service Provider*, SP) em nuvem (NASCIMENTO e WESTPHALL, 2011). O resultado foi a implantação de um Servidor Apache sobre uma máquina virtual contratada pelo Provedor de Nuvem *Amazon Web Services* - como ilustrado na Figura 19. Neste servidor, além da instalação do SP Shibboleth, uma aplicação foi escolhida para servir de exemplo de recurso a ser oferecido como serviço: o software de elaboração e edição colaborativa de documentos DokuWiki (DOKUWIKI, 2012). Os detalhes da implementação para o SP são apresentados a seguir.

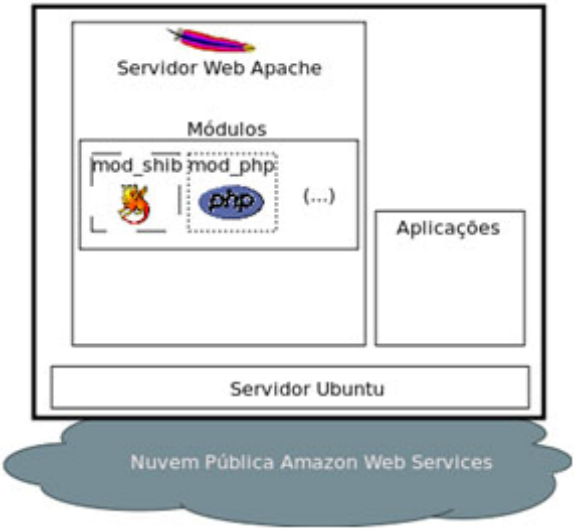

Figura 19. Diagrama do provedor de serviços implantado na nuvem.

Considerações iniciais e visão geral da implementação

O uso do software Eucalyptus foi estudado para a instalação do Provedor de Serviço Shibboleth hospedado em uma nuvem privada. No entanto, manutenções necessárias no sistema realizadas durante o período em que seria utilizado tornou impossível ser assim feito, e foi decidido que o software seria substituído pelos serviços da *Amazon Web Services* (AWS). O provedor de serviços Shibboleth foi instalado no serviço AWS e um protótipo de autenticação foi realizada através do uso do TestShib.

Para investigar o comportamento de um provedor de serviços Shibboleth sendo executado em uma máquina virtual hospedada em nuvem pública, foram utilizados os serviços do provedor de nuvem *Amazon Web Services* (AWS) (AMAZON, 2012), que oferece recursos computacionais sob demanda através da Internet. Além de ser um provedor amplamente utilizado por organizações em todo o mundo, e assim permitir que se teste o SP em um ambiente de nuvem real, outra motivação para a escolha deste serviço foi o fato de oferecer uma modalidade de conta gratuita para experimentá-lo. Entre outros provedores que poderiam ter sido utilizados para o propósito deste trabalho, citam-se o Microsoft Azure (MICROSOFT, 2012) e o Rackspace (RACKSPACE, 2012). A versão do SP instalada foi a 2.3.1, e a do servidor Apache 2.2.14.

Posteriormente uma aplicação foi instalada no provedor de serviços e configurada para utilizar a autenticação e autorização Shibboleth, a princípio com o mesmo TestShib, e depois com um provedor de identidades diferente.

Um primeiro teste de funcionamento da autenticação foi realizado com o uso do TestShib, que fornece um IdP próprio para se testar instalações do Shibboleth. A seguir uma aplicação mais elaborada foi instalada e configurada para ser protegida pelo Shibboleth. Por fim, o SP foi configurado para interagir com o Provedor de Identidades desenvolvido para o trabalho de (SANTOS e WESTPHALL, 2011).

O resultado final obtido pode ser ilustrado pela Figura 20: a aplicação protegida dentro de um provedor de serviços hospedado na nuvem é acessada pelo usuário quando ele digita seu endereço no navegador e o Shibboleth, através do módulo Apache `mod_shib`, redireciona o acesso para o provedor de identidades (IdP) onde é feita a autenticação e retornada uma asserção SAML com o resultado. Sendo autenticado com sucesso, os atributos liberados pelo IdP (e mapeados explicitamente pelo SP) são associados à variável de ambiente `REMOTE_USER`, e disponibilizados para a aplicação - que tem liberdade sobre o que fazer com eles, escolhendo como realizar a autorização.

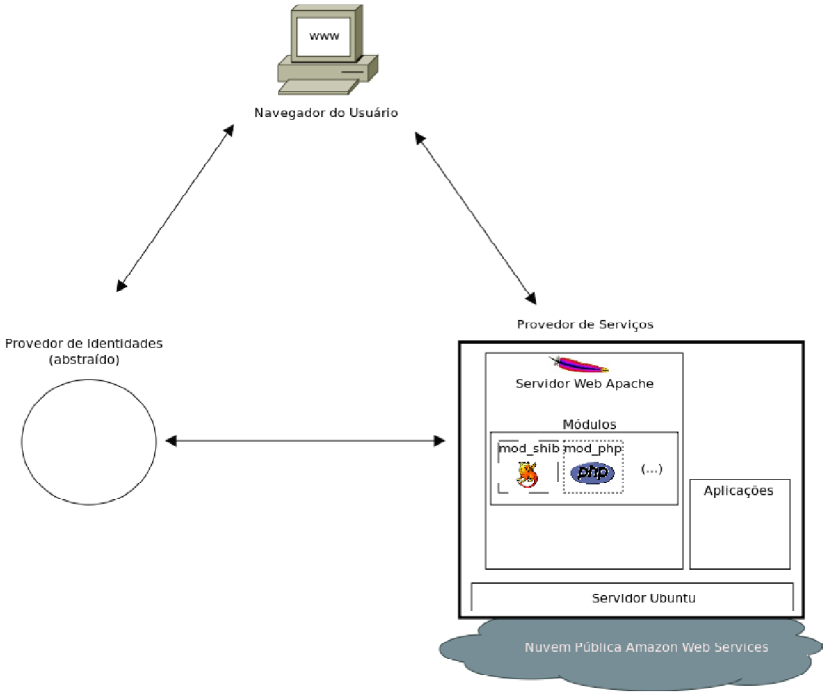

Figura 20. Diagrama simplificado do resultado final.

Criação de uma conta no *Amazon Web Services* (AWS) e primeiras configurações

A contratação dos serviços baseados em nuvem da Amazon é realizado a partir do endereço http://aws.amazon.com/, e o processo é bastante familiar para um usuário de Internet: insere-se o nome, o endereço de e-mail e alguns outros dados pessoais como endereço e telefone. Um contrato é disposto ao usuário com detalhes dos serviços, incluindo seus preços que em sua maioria são taxados por hora e cobrados ao final do mês. Para este exercício de criação de uma máquina virtual onde seria instalado o SP Shibboleth, foi utilizado o serviço *Amazon Elastic Compute Cloud* (EC2) - que fornece recursos computacionais sob demanda através de máquinas virtuais hospedadas na nuvem, e o *Amazon Elastic Block Store* (EBS) - para a persistência de dados do provedor de serviços independente da instância.

Após criada a conta e escolhidos os serviços a serem utilizados, pôde-se acessar o Console AWS (Figura 21) e proceder com os seguintes passos para a criação de uma instância de máquina virtual:

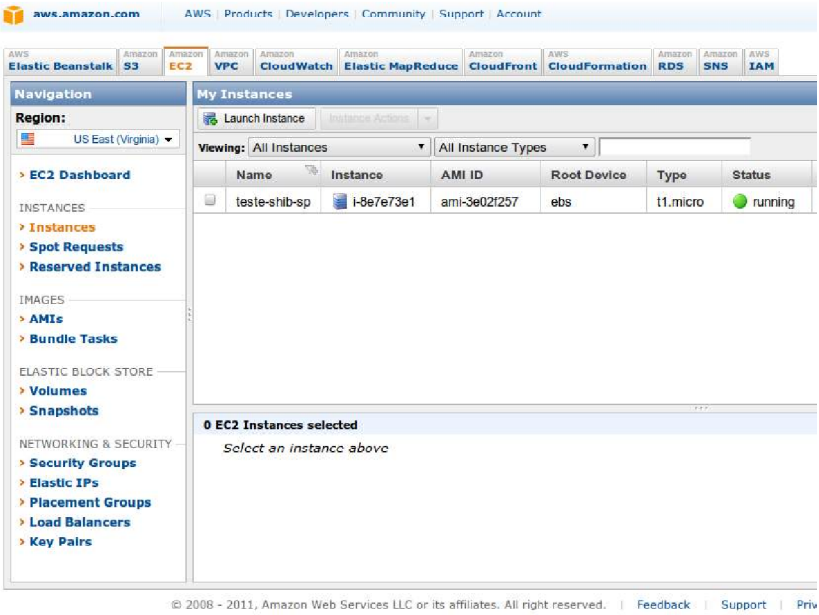

Figura 21. Console de Gerenciamento dos serviços da AWS com a instância já configurada.

1. Escolhe-se a região geográfica em que deseja-se que os dados estejam localizados fisicamente. A região determina o preço pago pelos serviços. Foi escolhida a região US-East;
2. Escolhe-se a imagem de sistema operacional (*Amazon Machine Image* ou AMI) utilizada para a instância, dentre uma lista com sistemas de diversos tipos e finalidades. Foi escolhido o sistema operacional Ubuntu 10.04 32 bits;
3. Escolhido o tipo de instância, que determina a capacidade de processamento inicial da máquina virtual, memória, capacidade de armazenamento e performance de operações de E/S. Uma instância de tipo micro foi utilizada, com até 2 núcleos virtuais de processamento e 613 GB de memória;
4. Depois de definidos alguns detalhes como o nome da máquina, foi criado um par de chaves criptográficas assimétricas utilizadas para o acesso via SSH;
5. Com a instância criada, foi possível configurar regras de firewall de dentro do próprio console através da criação de grupos de segurança, que permitem que se salvem regras de filtragem de pacotes de forma independente das instâncias. Para que a instância fosse utilizada como um servidor web, foi liberado o acesso às portas TCP 22, 80 e 443, correspondentes às permissões de acesso via SSH, HTTP e HTTPS, respectivamente.
6. Inicialmente, as instâncias não possuem um endereço IP estático - um endereço diferente é associado à ela toda vez que for instanciada - o que não é ideal para o uso do Shibboleth. Como alternativa o AWS oferece o serviço Elastic IP com um custo adicional, que permite não só que se mantenha um endereço IP fixo, mas que ele possa ser mapeado dinamicamente entre instâncias diferentes, de forma transparente ao usuário. A utilização deste recurso não foi necessária para os fins deste trabalho, sendo a máquina acessada pelo IP e *hostname* dinâmicos.
7. Após ter a instância criada e funcionando, para acessá-la via linha de comando bastou-se utilizar o comando ssh como exemplificado na Figura 22, onde “tj-sp-keys.pem” refere-se à chave privada criada anteriormente e “ec2-174-129-156-69.compute-1.amazonaws.com” ao *hostname* associado automaticamente à instância no momento de criação, e que foi utilizado neste trabalho como definitivo para identificação da máquina.

```
ssh -i tj-sp-keys.pem ubuntu@ec2-174-129-156-69.compute-
1.amazonaws.com
```

Figura 22. Comando ssh para acessar a instância.

Instalação do Provedor de Serviços (SP) Shibboleth

Uma vez conectado à instância de máquina virtual, pôde-se iniciar a instalação do SP Shibboleth. Os passos iniciais foram:

1. Configuração dos arquivos "/etc/hostname" e "/etc/hosts" com o endereço de *host* e o IP correto da instância obtidos pelo console de gerenciamento do AWS;
2. Criação de chaves e certificado para o servidor Apache e o SP Shibboleth através da ferramenta *openssl*;
3. Instalação do servidor Apache e seus módulos para PHP e Shibboleth, através dos pacotes apache2, libapache2-mod-php5 e libapache2-mod-shib2, disponíveis nos repositórios oficiais do Ubuntu;
4. Liberação do Apache permitindo conexões às portas 80 e 443 com a ferramenta *iptables*;
5. Configuração do Virtual Host padrão do Apache, para conexões na porta 80, com a alteração do arquivo /etc/apache2/sites-available/default descrita na Figura 23;

```
<VirtualHost *:80>
    ServerName ec2-174-129-156-69.compute-1.amazonaws.com
    (...)
   DocumentRoot /var/www/
    <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride None
            Order allow,deny
            allow from all
    </Directory>
   (...)
</VirtualHost>
```

Figura 23. Conteúdo do arquivo /etc/apache2/sites-available/default.

6. Configuração do Virtual Host da página que será protegida pelo Shibboleth, em conexões pela porta 443 e associada ao diretório "/secure/", com a alteração do arquivo "/etc/apache2/sites-available/shibboleth-sp2" (Figura 24).

```
<VirtualHost *:443>
    ServerName ec2-174-129-156-69.compute-1.amazonaws.com
    (...)
    <Location /secure>
            AuthType shibboleth
            ShibRequireSession On
            require valid-user
            Order allow,deny
            allow from all
     </Location>
    (...)
</VirtualHost>
```

Figura 24. Conteúdo do arquivo "/etc/apache2/sites-available/shibboleth-sp2".

Com estas configurações iniciais sendo concluídas, já foi possível ter o Apache instalado e funcionando para conexões desprotegidas. As modificações seguintes foram necessárias para que o Shibboleth fosse utilizado para a autenticação:

1. Ativação dos módulos Apache do Shibboleth, SSL e Rewrite através dos comandos listados na Figura 25;

```
a2enmod shib2
a2enmod ssl
a2enmod rewrite
```

Figura 25. Comandos para a ativação de módulos do servidor Apache.

2. Ativação do site protegido "shibboleth-sp2" pelo comando `a2ensite shibboleth-sp2`;
3. Configuração dos atributos de usuários que deseja-se obter do IdP após a autenticação: O provedor de serviços só considera atributos que estejam devidamente declarados no

arquivo “/etc/shibboleth/attribute-map.xml”. Foi utilizado o mapeamento fornecido pelo tutorial do site da RNP, que utiliza o `brEduPerson` - um esquema proposto para a definição de atributos de usuários membros de comunidades acadêmicas brasileiras (RNP, 2008);

4. Definição de filtros adicionais para a obtenção e armazenamento de determinados atributos, baseados em regras, através da modificação do arquivo “/etc/shibboleth/attribute-policy.xml”.
5. Configuração do arquivo de metadados do provedor de serviços (Figura 26), necessário para a identificação de cada membro de uma Federação, localizado em “/root/ec2-174-129-156-69-metadata-sp.xml”. Neste arquivo é definido o atributo identificador o qual outros membros da federação irão usar para se referir ao SP, assim como o certificado X.509 criado anteriormente, dados organizacionais e informações para contato. Outro elemento importante é o `<AssertionConsumerService>` que define os locais (associados a URI's) responsáveis por processarem Asserções SAML via determinados protocolos.

```
<AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST-
SimpleSign" Location="https://ec2-174-129-156-69.compute-
1.amazonaws.com/Shibboleth.sso/SAML2/POST-SimpleSign"
index="2"/>

      <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
Artifact" Location (...)

      <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:PAOS"
(...)

      <AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:1.0:profiles:browser-
post" (...)
```

Figura 26. Elementos `<AssertionConsumerService>` no arquivo de metadados do SP.

Ao finalizar esta etapa, o provedor de serviço foi configurado para se comunicar com um IdP específico, de forma a autenticar usuários para acessar seu conteúdo.

Teste de funcionamento da autenticação com o TestShib

Concluída a instalação do Provedor de Serviço na instância AWS foi possível testar seu funcionamento com a utilização do serviço TestShib, disponível a partir do endereço http://www.testshib.org/testshib-two/index.jsp.

Foi necessária a criação de uma conta em um dos IdPs registrados com o Testshib, sendo aqui utilizado o openidp, acessível em https://www.openidp.org/. Este registro permite que se utilize a autenticação do IdP para acessar Provedores de Serviços cadastrados. O provedor, por sua vez, é registrado separadamente por um formulário onde deve-se inserir dados como *hostname* do servidor, conteúdo de seu certificado e dados para contato. Este formulário é acessível em http://www.testshib.org/testshib-two/menu.do.

Através do site pode-se, então, criar automaticamente o arquivo de configuração shibboleth2.xml composto especificamente para a conexão com o Testshib. Ele deve ser copiado ao diretório "/etc/shibboleth/" do provedor de serviço. Este arquivo contém a maior parte da configuração necessária para o funcionamento do SP, e seu estudo nesta etapa foi importante para a compreensão do Shibboleth. Alguns elementos que valem ser observados (SHIBBOLETH, 2011):

`<RequestMapper>` - relaciona requisições enviadas ao SP com configurações definidas. A configuração utilizada (Figura 27) mapeia requisições feitas à URI https://ec2-174-129-156-69.compute1.amazonaws.com/secure para que seja imposta uma sessão autenticada, isto é feito relacionando as requisições a configurações definidas no elemento `<ApplicationDefaults>` através do atributo `applicationID`;

```
<RequestMapper type="Native">
     <RequestMap applicationId="default">
       <Host name="ec2-174-129-156-69.compute-
1.amazonaws.com">
                 <Path name="secure" authType="shibboleth"
       requireSession="true"/>
       </Host>
     </RequestMap>
</RequestMapper>
```

Figura 27. Elemento `<RequestMapper>` do arquivo shibboleth2.xml.

<ApplicationDefaults> - aplicações neste contexto representam coleções distintas de recursos que compartilham necessidades de configurações específicas. Dentro deste elemento é definido a maior parte do comportamento do provedor de serviços Shibboleth sobre estes recursos. A Figura 28 detalha o elemento, onde observa-se que a aplicação utilizada chama a política de segurança de manuseio de atributos default (como definido no attribute-policy.xml) e associa à variável de ambiente REMOTE_USER o atributo EduPersonPrincipalName (eppn) do usuário;

```
<ApplicationDefaults id="default" policyId="default"
REMOTE_USER="eppn" entityID="https://ec2-174-129-156-
69.compute-1.amazonaws.com/shibboleth-sp"

homeURL="https://ec2-174-129-156-69.compute-
1.amazonaws.com/index.html">
```

Figura 28. Elemento <ApplicationDefaults> do arquivo shibboleth2.xml.

<MetadataProvider> - define a fonte de metadados para uso do provedor de serviço, ou seja, através do qual ele descobre os detalhes necessários para se comunicar com o IdP. A configuração deste elemento para funcionar com o TestShib é observado na Figura 29;

```
<MetadataProvider type="XML"
uri="http://www.testshib.org/metadata/testshib-
providers.xml" backingFilePath="testshib-two-idp-
metadata.xml" reloadInterval="180000" />
```

Figura 30. Elemento <MetadataProvider> do arquivo shibboleth2.xml.

<Sessions> - controla o comportamento das sessões estabelecidas, seu tempo de vida, tempo necessário de inatividade para encerrá-la, entre outras opções (Figura 30);

```
<Sessions lifetime="28800" timeout="3600"
checkAddress="false" handlerURL="/Shibboleth.sso"
handlerSSL="false">
```

Figura 29. Elemento <Sessions> do arquivo shibboleth2.xml.

<AssertionConsumerService> - definição do ponto de entrada onde serão recebidas asserções SAML para iniciar sessões (Figura 31);

```
<md:AssertionConsumerService Location="/SAML2/POST"
index="1"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
POST"/>
```

Figura 31. Elemento <AssertionConsumerService> do arquivo shibboleth2.xml.

Tendo realizado a configuração do SP para aceitar o acesso autenticado pelo TestShib, foi possível testar o funcionamento do processo acessando o endereço http://ec2-174-129-156-69.compute-1.amazonaws.com/secure, e observar que o navegador é redirecionado para o site de autenticação do TestShib (Figura 32), e após inseridos nome de usuário e senha uma tela é apresentada com os atributos fictícios da conta utilizada, como observado na Figura 33.

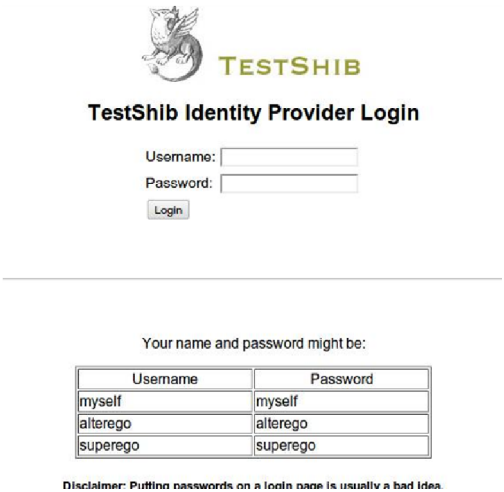

Figura 32. Página de autenticação TestShib.

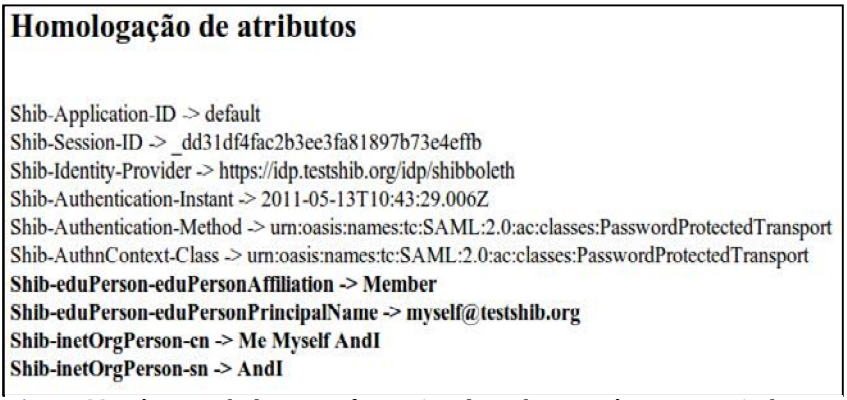

Figura 33. Página exibida com informações de atributos após autenticação bem-sucedida. Aplicação teste encontrada em **(RNP, 2011)**.

O TestShib permite ainda que se visualize os logs do IdP para acompanhar o processo de autenticação, como observado na Figura 34, onde podemos ver um exemplo de resposta do IdP a um processo de autenticação bem sucedido, devolvendo uma asserção criptografada (<saml2:EncryptedAssertion>) que conteria os atributos.

```
<saml2p:Response
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
Destination="https://ec2-174-129-156-69.compute-
1.amazonaws.com/Shibboleth.sso/SAML2/POST"
ID="_9830e4625f59f354ff7ea57268852bea"
InResponseTo="_209ec3a979290ab297b11d266f0dc384"
IssueInstant="2011-05-13T10:43:29.067Z" Version="2.0">

   <saml2:Issuer
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:entity">https://idp.testshib.org/idp/shibboleth</s
aml2:Issuer>

   <ds:Signature
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

      (...)

   </ds:Signature>

   <saml2p:Status>

      <saml2p:StatusCode
Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>

   </saml2p:Status>

   <saml2:EncryptedAssertion
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">

      (...)

   </saml2:EncryptedAssertion>

</saml2p:Response>
```

Figura 34. Trecho de uma asserção de resposta do IdP a um processo de autenticação bem sucedido.

Autenticação e Autorização Shibboleth em uma aplicação

Para se observar o funcionamento da autenticação e autorização Shibboleth em uma aplicação real, foi escolhido o programa Dokuwiki

(DOKUWIKI, 2012), um software código aberto criado por Andreas Gohr para a criação colaborativa de documentos (os chamados *wikis*).

Para a instalação, seguiu-se os passos encontrados no site do programa, que resumem-se em:

1. Baixar o pacote *dokuwiki* dos repositórios oficiais do Ubuntu;
2. Reiniciar o Apache;
3. Executar um script instalador fornecido pelo programa que realiza as configurações básicas automaticamente.

Através desta etapa o software foi instalado, mas ainda não disponibilizado para acesso externo ou protegido pelo Shibboleth. Para tal, foram necessárias duas configurações importantes:

1. A criação do arquivo "dokuwiki.conf" no local "/etc/apache2/conf.d/" de forma que seu conteúdo ficasse como na Figura 35. Estas diretivas associam o diretório onde o programa foi instalado com o endereço URL "<hostname>/dokuwiki/" , para que as páginas ali colocadas pudessem ser acessadas pelo navegador do usuário do serviço. Também é definida a proteção destas páginas pelo Shibboleth, mas sem a necessidade de uma Sessão imediatamente, como explicado a seguir;

```
Alias /dokuwiki /usr/share/dokuwiki


<Directory /usr/share/dokuwiki/>
       Options +FollowSymLinks
       AllowOverride All
       order allow,deny
       Allow from all
     AuthType shibboleth
       require shibboleth
</Directory>
```

Figura 35. Conteúdo do arquivo dokuwiki.conf.

2. A modificação do arquivo "/etc/shibboleth/shibboleth2.xml", especificamente o elemento `<RequestMapper>` que define os recursos a serem protegidos. A forma utilizada encontra-se na Figura 36. Com esta configuração, além de se instruir o Shibboleth a proteger os recursos dentro de

"/dokuwiki/", utiliza-se o mecanismo de *lazy session*, ou seja, permite-se o acesso a uma pagina inicial sem uma sessão ser iniciada - sem autenticação - e dá-se a liberdade à aplicação de decidir quando deve ser iniciada uma Sessão segura.

```
  <RequestMapper type="Native">
        <RequestMap applicationId="default">
            <Host name="ec2-174-129-156-69.compute-
1.amazonaws.com">
                <Path name="dokuwiki"
requireSession="false" authType="shibboleth" />
            </Host>
        </RequestMap>
   </RequestMapper>
```

Figura 36. Conteúdo do elemento `<RequestMapper>` no arquivo shibboleth2.xml.

Com estas opções, um usuário que acesse a página inicial do *wiki* não precisa imediatamente autenticar-se, tendo acesso de leitura ao conteúdo. Para incluir, editar ou apagar conteúdo textual do serviço, porém, ele deve autenticar-se clicando em um botão específico (*login*) da página inicial que serve de gatilho para o processo de autenticação com o provedor de identidades (IdP) especificado nas configurações.

Antes de liberar acesso aos usuários, foi necessário ainda que se especificasse quais atributos, dentro daqueles liberados pelo IdP, seriam utilizados pela aplicação e como eles seriam utilizados. Esta etapa é específica da aplicação, e é demonstrada na Figura 37, pelo conteúdo do arquivo "/etc/dokuwiki/local.php", que associa os atributos do IdP "Shib-inetOrgPerson-cn" e "Shib-eduPersonPrincipalName" aos atributos da aplicação "var_remote_user" e "var_name", respectivamente. Outras associações seriam possíveis.

```
$conf['auth']['shib']['var_remote_user'] = 'Shib-
inetOrgPerson-cn';

$conf['auth']['shib']['var_name'] = 'Shib-eduPerson-
eduPersonPrincipalName';
```

Figura 37. Conteúdo do arquivo /etc/dokuwiki/local.php.

Realizada a autenticação, os processos de autorização dentro do provedor de serviços Shibboleth podem ser realizados de três formas:

- via diretivas no arquivo ".htaccess" do Apache, de forma similar à configuração demonstrada na Figura 35, onde instruções "require" podem incluir usuários específicos, grupos, etc;
- via elemento `<AccessControl>` dentro do elemento `<Path>` ilustrado na Figura 36. Este elemento oferece diversas possibilidades para casos de uso mais complexos de controle de acesso (SHIBBOLETH, 2011);
- via aplicação, que tem liberdade para criar regras internas de acordo com os atributos a ela acessíveis.

Para se diferenciar usuários comuns de usuários administradores do Dokuwiki, foi testada a autorização via aplicação através da instrução dentro do arquivo "/etc/dokuwiki/local.php" mostrada na Figura 38, que declara uma lista de usuários a serem considerados superusuários (*superusers)*: aqueles com nome de usuário "admin" e "user99". Esta configuração também é específica da aplicação.

```
$conf['auth']['shib']['superusers'] = array( 'admin' , 'user99');
```

Figura 38. Instrução para declaração de *superusers*, dentro do arquivo /etc/dokuwiki/local.php.

Finalizadas estas modificações, foi possível realizar o acesso autenticado e autorizado na aplicação com sucesso. O resultado é apresentado na Figura 39.

Figura 39. Página inicial do *Wiki* protegido pelo Shibboleth.

4.3.2 Instalação do provedor de identidades (IdP) Shibboleth

O IdP implementado teve como foco a privacidade dos usuários de serviços na nuvem. Por este motivo, a proposta do trabalho é a utilização de uma aplicação que englobe um provedor de identidades com uma camada de proteção à privacidade executando num ambiente de nuvem. Uma visão geral da proposta pode ser visualizada na Figura 40 e é descrita logo a seguir:

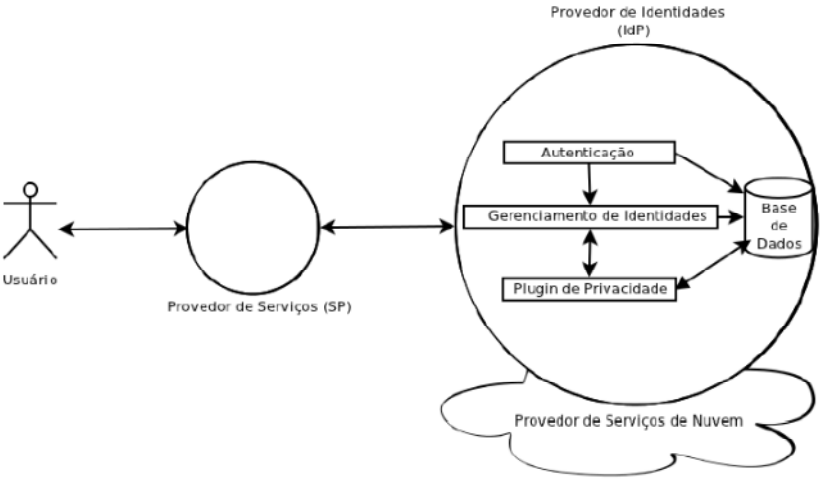

Figura 40. Diagrama geral da proposta.

No cenário desta proposta o usuário interessado em utilizar um serviço protegido acessa diretamente o provedor de serviços, o provedor de serviços então o redireciona para seu respectivo provedor de identidades, que pode ser informado pelo usuário e deve ter a confiança do provedor de serviços. O provedor de identidades está executando em um ambiente de nuvem, o que é transparente para o usuário. O provedor de identidades pede a autenticação do usuário e acessa seus atributos em sua base de dados. Quando o usuário está autenticado e antes de ser novamente redirecionado para o provedor de serviços seus dados passam por um *plugin* de privacidade, onde o usuário fica ciente e deve consentir com a liberação de seus atributos.

Instalação da Infraestrutra Básica

A instalação da aplicação começou com a infra-estrutura básica que seria utilizada e o primeiro passo foi definir o serviço de nuvem em que ele estaria disponível. A aplicação deveria utilizar uma Infraestrutura como um Serviço (IaaS) e ter algumas características essenciais como possibilidade de persistência dos dados e possibilidade de utilização de IPs estáticos, além da possibilidade de liberação de portas no firewall.

Optou-se então pelo uso de uma nuvem pública. O serviço escolhido foi o EC2 da *Amazon Web Services* (AWS). Foi instanciada uma máquina virtual executando Windows Server 2008 (*Amazon Machine Image* (AMI) ID ami-c3e40daa) e utilizado o serviço *Elastic IP* para obter um IP estático para essa instância. O IP obtido foi o 50.19.108.64, com *Domain Name System* (DNS) público ec2-50-19-108-64.compute-1.amazonaws.com. Para persistência dos dados utilizou-se um volume EBS de 30GB.

As portas liberadas no firewall foram: 3306 para acesso ao banco de dados MySQL, 3389 para acesso remoto via Remote Desktop Protocol (RDP), 8009 para uso do Shibboleth e 8080 para uso do Tomcat.

Com a máquina instanciada e em execução começou-se a instalação da aplicação. Primeiramente foi instalado o servidor web Apache, versão 2.2 (The Apache Software Foundation, 2011a). Foi gerado um certificado digital para a máquina e o servidor Apache foi configurado para permitir o uso de *Secure Sockets Layer* (SSL) com esse certificado. O servidor foi configurado para aceitar na porta 80 conexões não-SSL e nas portas 443 e 8443 conexões SSL.

Depois foi instalado o *Java Development Kit* (JDK), versão 1.6.0 25 (Oracle Corporation, 2011a), e finalmente o servidor de aplicações Apache Tomcat 6.0.22 (The Apache Software Foundation, 2011b), no qual deveriam ser executadas as aplicações de autenticação, gerenciamento de identidades e o *plugin* de privacidade.

Foi então configurado um *proxy* no Apache para repassar os pedidos dessas aplicações para o Tomcat. Isso foi feito com a criação de um arquivo “httpd-mod-proxy.conf”.

O primeiro serviço a ser instalado foi o mecanismo de autenticação. O serviço escolhido foi o JASIG CAS *Server* (JASIG, 2010), versão 3.3.2, que realiza a autenticação de usuários através de *login* e senha e possibilita *Single Sign On* através de uma interface web e então repassa os usuários autenticados para o Shibboleth. O CAS foi

configurado para procurar os usuários em um diretório *Lightweight Directory Access Protocol* (LDAP).

Para utilizar esse diretório foi instalado o OpenLDAP (OPENLDAP FOUNDATION, 2012) em uma outra máquina virtual, um Ubuntu 10.10 (AMI ID ami-cef405a7) também executando na nuvem da Amazon.

Instalação do Shibboleth

Com esses passos iniciais prontos foi possível começar a instalação do provedor de identidades Shibboleth. Primeiramente a aplicação do IdP em si foi configurada e instalada no Tomcat.

O Shibboleth precisa fazer parte de uma federação para que possa ser utilizado com provedores de serviço. A federação escolhida para ser utilizada nesse trabalho foi a TestShib (INTERNET2, 2011), criada para propósito de testes de configurações do Shibboleth, tanto de SPs quanto de IdPs. Para utilizar o TestShib foi necessário cadastrar o IdP, informando o endereço DNS e o certificado gerado anteriormente. O Shibboleth foi então configurado para utilizar os metadados do TestShib. Nesse arquivo também foi configurado o caminho do certificado gerado anteriormente e utilizado pelo Shibboleth. Para que o Shibboleth recebesse a autenticação do CAS Server foi utilizado o CAS *Client*.

Com esses passos prontos o Shibboleth está corretamente instalado e autenticando usuários a partir do CAS. No entanto o provedor de identidades ainda não liberava nenhum atributo de usuário para o provedor de serviços, o que tornaria seu uso inviável para a maioria das aplicações práticas.

Na liberação dos atributos de usuário é que reside a maior preocupação com a privacidade no gerenciamento de identidades, pois são os atributos que contem os dados sensíveis dos usuários e que devem ser tratados com cuidado na hora em que são criados, armazenados, transferidos ou destruídos.

Os atributos dos usuários seguem um esquema que é definido no diretório LDAP. O esquema mais comum é o `eduPerson` (INTERNET2, 2012), que contém atributos comuns à membros de uma federação acadêmica. O esquema utilizado neste trabalho foi o `brEduPerson` (RNP, 2010), uma extensão do `eduPerson` para federações brasileiras.

Alguns exemplos de atributos presentes nos esquemas `eduPerson` e `brEduPerson` e que foram utilizados nestes trabalho

são: `eduPersonPrincipalName`, geralmente representado pelo nome de usuário utilizado no *Single Sign On*, esse atributo é utilizado quando se necessita um identificador persistente através de vários serviços; `eduPersonAffiliation`, representa o tipo de afiliação de um usuário com a federação da qual faz parte e `brPersonCPF`, o número do CPF do usuário.

A configuração da liberação dos atributos dos usuários é feita em dois arquivos: "attribute-resolver.xml" e "attribute-filter.xml". O primeiro define cada atributo que pode ser liberado pelo IdP e de onde ele é recuperado. O segundo define a política de liberação de atributos do provedor de identidades, tanto de maneira geral quanto de maneira específica para cada atributo.

No primeiro foram criadas definições para quatro atributos que seriam utilizados como exemplo: `eduPersonPrincipalName`; `eduPersonAffiliation`; `brPersonCPF`, o CPF do usuário; cn, o primeiro nome do usuário e sn, o sobrenome do usuário, conforme definidos no esquema `brEduPerson`. Com exceção do atributo `eduPersonAffiliation`, que foi definido como um atributo estático, sempre retornando o valor "member", todos os outros são pesquisados em um diretório LDAP.

No segundo foram criadas regras de liberação para cada um dos atributos. Como esse provedor de identidades só seria utilizado com o provedor de serviços do TestShib e apenas para fins de teste, sendo o objetivo principal do trabalho demonstrar o uso da camada de privacidade, os atributos podem ser liberados para qualquer provedor de serviço, o que não seria seguro em um ambiente de produção.

Com essas configurações o Shibboleth está pronto para ser usado com um provedor de serviços que requisite a liberação de atributos. O próximo passo é a instalação do *plugin* de privacidade.

Instalação do Uapprove

O uApprove é um *plugin* de privacidade para o Shibboleth desenvolvido pela rede de universidades suíças SWITCH, para uso em sua federação acadêmica, a SWITCHaai. A versão 2.2.1 foi utilizada neste trabalho.

De acordo com (SWITCH, 2012) o uApprove serve aos seguintes propósitos:

1. Para o usuário, implementa um mecanismo que o informa sobre a liberação de atributos para um provedor de serviços;

2. Para o administrador de um provedor de identidades, provê uma ferramenta para implementar leis de proteção à privacidade exigindo o consentimento dos usuários antes da liberação dos atributos e permite coletar informações sobre essa liberação e acessos a um provedor de serviços.

O objetivo do uApprove é garantir que o usuário saiba quais dados seus são liberados e para quem são liberados, além disso o usuário deve concordar com os termos de uso do provedor de serviços.

Para a instalação do uApprove é necessário utilizar uma base de dados que armazene informações sobre o consentimento dos usuários e a liberação de seus atributos. Para esse armazenamento foi utilizado o sistema de gerenciamento de bancos de dados MySQL, versão 5.5 (ORACLE, 2012), instalado na mesma máquina do Shibboleth. Primeiramente foi criado um banco de dados e um usuário para o uApprove. E em seguida foram criadas as tabelas a serem utilizadas pelo *plugin*.

Foi então gerado um arquivo “terms-of-use.xml” que contém um exemplo de Termos de Uso. Também é necessário definir um segredo compartilhado que será utilizado para criptografar mensagens transmitidas entre o IdP *plugin* e o Viewer.

Instalação Concluída

Com a instalação concluída, uma visão geral da aplicação pode ser resumida na Figura 41:

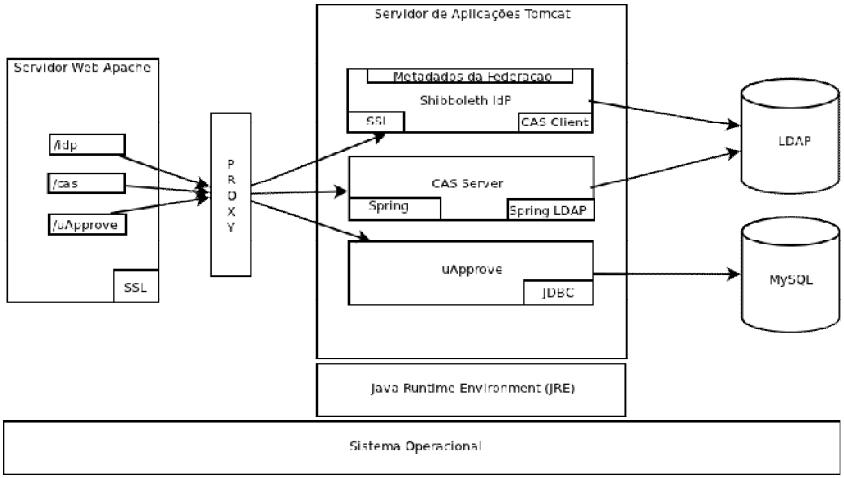

Figura 41. Diagrama detalhado do IdP.

A Figura 41 representa a visão detalhada da parte do IdP que havia sido descrita na Figura 40. Como ponto de acesso temos o servidor web apache, que recebe as requisições *Hypertext Transfer Protocol Secure* (HTTPS). O Apache tem configurado um proxy que encaminha essas requisições para o Tomcat, para que sejam recebidas pela aplicação correta. Dentro do Tomcat existem três aplicações sendo executadas:

Shibboleth IdP Configurado com os metadados da federação e utilizando o CAS *client* para receber a autenticação, é acessado através de "/idp";

CAS Server Executa a autenticação e a envia para o IdP. Utiliza o framework web Spring e é acessado através de "/cas";

uApprove Obtém o consentimento do usuário e libera os atributos para o IdP. Acessa o banco de dados através da *Application Programming Interface* (API) *Java Database Connectivity* (JDBC) e é acessado através de "/uApprove".

O IdP e o CAS *Server* obtém os dados dos usuários através de um diretório LDAP e o uApprove utiliza o banco de dados MySQL para registrar as informações de consentimento dos usuários.

4.3.3 Instalação do segundo provedor de identidades (IdP) Shibboleth

Para demonstrar o uso do SP por mais de um cliente, outro IdP foi implementado, também em nuvem, similar ao primeiro. A partir deste ponto, o conceito *multi-tenancy* se faz necessário, uma vez que o serviço disponibilizado pelo SP será compartilhado por vários clientes. Para dar apoio a esta tarefa o Shibboleth disponibiliza de um componente chamado WAYF (*Where Are You From*), também chamado de Serviço de Descoberta (*Discovery Service*) que é responsável por permitir a associação entre um usuário e uma organização. Este mecanismo foi configurado junto ao SP para gerenciar as instituições pertencentes à federação.

Este segundo IdP ficou mais simplificado por não tratar especificamente os problemas de privacidade que foram tratados no primeiro. Todo o processo de implementação usado no primeiro IdP foi usado para este segundo. O resultado pode ser visto na Figura 42.

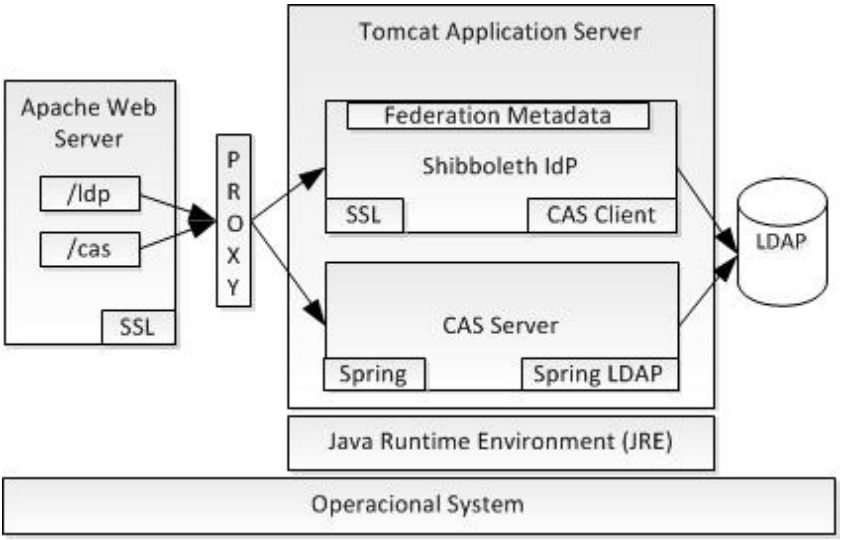

Figura 42. Diagrama detalhado do IdP.

4.3.4 Integração entre os provedores

Após a criação de cada provedor (o de serviço e mais os outros dois IdPs), foi realizada a integração entre eles, para que pudessem estar dentro de uma mesma federação e compartilhar recursos.

Primeiro foi realizada a integração do SP com o IdP I. Para tanto foi necessário substituir, no SP, o IdP do TestShib pelo IdP I, realizando os seguintes passos:

1. Especificar o novo endereço de onde se obter o arquivo de Metadados do IdP que, como mencionado anteriormente, serve para especificar todos os detalhes do provedor de identidades para que ambos os servidores se comuniquem corretamente. Isto foi feito modificando-se o elemento `<MetadataProvider>` no arquivo shibboleth2.xml (Figura 43);

```
<MetadataProvider type="XML" uri="https://ec2-50-19-108-
64.compute-1.amazonaws.com/idp-metadata.xml"


                backingFilePath="/etc/shibboleth/idp-
metadata.xml" validate="true" reloadInterval="300"/>
```

Figura 43. Elemento `<MetadataProvider>` no arquivo shibboleth2.xml.

2. Especificar o novo IdP a ser direcionado assim que uma nova Sessão é iniciada. Isto foi feito modificando-se o atributo `entityId` do elemento `<SessionInitiator>` também do arquivo shibboleth2.xml. Este `entityId` refere-se ao elemento dentro do arquivo de Metadados (mencionado no passo anterior) onde se busca as informações necessárias para a autenticação. O conteúdo final deste elemento encontra-se na Figura 44.

```
<SessionInitiator type="SAML2" Location="/Login"
isDefault="true" defaultACSIndex="1" id="idp-daniel"

                entityID="https://ec2-50-19-108-
64.compute-1.amazonaws.com/idp/shibboleth"
template="bindingTemplate.html" />
```

Figura 44. Elemento `<SessionInitiator>` no arquivo shibboleth2.xml.

4.4 CASOS DE USO: ANÁLISE E RESULTADOS DE TESTES DENTRO DO CENÁRIO

O resultado da implantação dos provedores de identidade e de serviço é representado na Figura 45.

Nesta estrutura resultante cada IdP é representado em uma nuvem privada, e a nuvem onde se encontra o SP representa uma nuvem pública.

Uma vez o cenário implantado, alguns testes foram realizados. Sendo destacados os resultados de dois casos de uso principais:

Figura 45. Resultado da implantação do cenário

4.4.1 Acesso para leitura de documentos

No caso de acesso somente leitura, o serviço oferecido possibilita o uso anônimo. Para realizar este tipo de acesso, basta que o usuário digite o URL do serviço desejado em seu navegador web. Neste caso, o serviço está também disponível para acessos externos à federação.

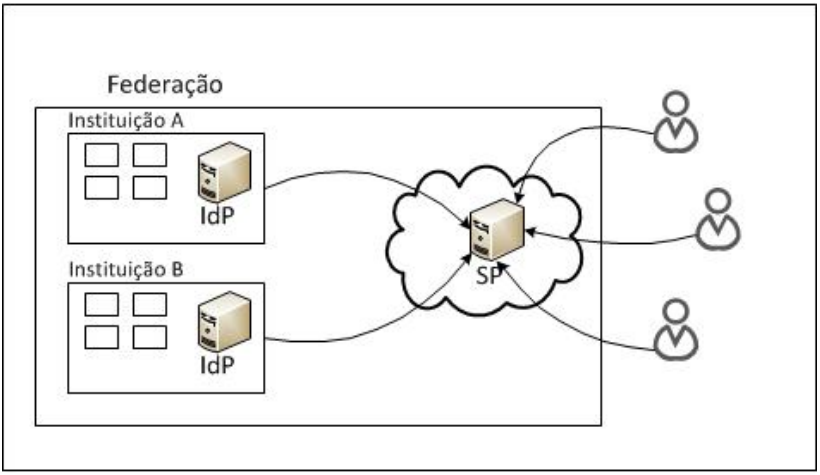

Figura 46. Caso de uso de acesso aos serviços disponibilizados publicamente.

Conforme mostra a Figura 46, o provedor de serviços não necessita se comunicar com o provedor de identidades. O serviço neste caso é oferecido ao público em geral e dispensa a necessidade de autenticação.

Nesta figura é representado o acesso ao provedor de serviços por usuários que fazem parte da federação (Instituições A e B), bem como de usuários externos à federação.

4.4.2 Acesso para leitura, alteração e gravação de documentos

Para que um usuário tenha permissão para alterar documentos, então se faz necessário sua autenticação. Nesse caso, o usuário irá solicitar a autenticação através do botão "*login*" que o encaminha à uma URL protegida pelo *Shibboleth*. O processo mod_shib verifica que o usuário não tem uma sessão aberta e o encaminha para o serviço de descoberta da federação (WAYF), onde o usuário escolhe sua instituição e é encaminhado para a URL do Provedor de Identidades (IdP), da instituição escolhida, através de redirecionamento HTTP. Na página web do IdP, o usuário insere suas credenciais (nome e senha) e caso bem-sucedido, *cookies* são registrados e um *handle* - uma asserção SAML com os dados da autenticação - é criado.

Este *handle* é utilizado pelo SP para requisitar os atributos do usuário ao IdP, que analisa o pedido com base em regras de liberação previamente estabelecidas. Se o *handle* for válido e a requisição aceita, outro *cookie* é criado e o usuário é finalmente redirecionado ao SP (ao aplicativo web que acessou originalmente), e seus atributos são enviados pelo módulo *Shibboleth* a este aplicativo como valores de variáveis de ambiente, para que ele possa usá-lo da forma que escolher. Através destes atributos o aplicativo emprega regras internas de autorização, para determinar se o usuário tem direitos administrativos sobre o sistema.

A Figura 47 mostra em linhas gerais a comunicação realizada entre o IdP e o SP no momento em que o acesso restrito é solicitado. Pode-se notar que o acesso por um usuário qualquer, que esteja fora do domínio federado, não é possível.

A comunicação entre SP e IdP é iniciada quando um usuário, pertencente a qualquer instituição, que faça parte da federação, requer o uso de um recurso ou serviço com restrições de acesso. Na figura o funcionamento interno tanto do SP quanto do IdP foi abstraído.

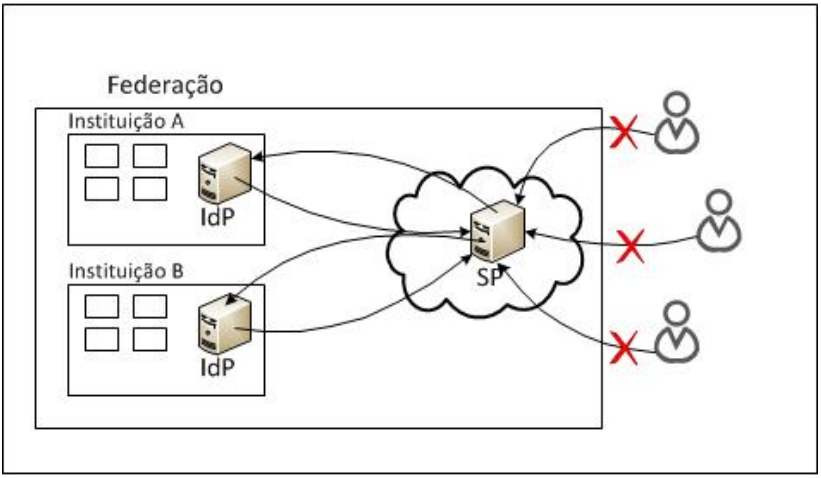

Figura 47. Caso de uso de acesso restrito a recursos e/ou serviços.

5 CONCLUSÃO

A adoção do gerenciamento de identidades segura em um ambiente de nuvem aborda as questões de provisionamento de identidades, autenticação, autorização e federação. O uso de federações no gerenciamento de identidades tem um papel vital ao permitir que organizações autentiquem seus usuários de serviços de nuvem usando qualquer IdP escolhido (CSA, 2009).

Para a compreensão sobre as questões de segurança em um cenário de nuvem e realização deste trabalho, foi necessário o estudo detalhado sobre computação em nuvem, gerenciamento de identidades e padrões de trocas de mensagens para controle de acesso, como SAML, bem como, compreender os desafios de segurança inerentes a um ambiente de nuvem.

Além disso, ficou claro que o uso de federações de identidades é fundamental para solucionar os problemas encontrados em relação ao controle de acesso nas nuvens computacionais. Através das federações é possível simplificar o acesso dos usuários aos recursos, uma vez que promovem a autenticação única (*Single Sign-On*). As federações também facilitam o compartilhamento de recursos entre organizações diferentes, promovendo, assim, a cooperação entre os participantes de um círculo de confiança.

A ferramenta Shibboleth permite a construção de federações e possui grandes facilidades no que diz respeito à configuração do controle de acesso, seja num ambiente comum, ou num ambiente de nuvem. É uma ferramenta flexível e pode se adaptar aos mais diversos cenários.

Por outro lado, pode ser considerada, inicialmente, uma ferramenta complexa. Consequentemente, é necessária uma análise detalhada sobre seu funcionamento e estrutura.

5.1 CONTRIBUIÇÕES

O trabalho (ALBESHRI e CAELLI, 2010) apresenta apenas um framework teórico. O trabalho de (RANCHAL, BHARGAVA, *et al.*, 2010) desenvolveu uma forma de proteger a privacidade em ambiente de nuvem e fez um protótipo usando a tecnologia de agentes Java no ambiente JADE. Usa um pacote ativo (*active bundle*) que é um

recipiente (container) composto por dados sensíveis, metadados e uma máquina virtual. Em (ANGIN, BHARGAVA, *et al.*, 2010) os autores propõem os mecanismos criptográficos usados no trabalho de (RANCHAL, BHARGAVA, *et al.*, 2010), sem qualquer tipo de implementação ou validação. Na Tabela 1 é apresentado um quadro comparativo dos trabalhos.

Tabela 1. Quadro comparativo com os trabalhos relacionados.

	Identidades Federadas	**Centrado no Usuário**	**Identificação anônima**	**Ambiente em Nuvem**	**Implementação**
Este trabalho	X			X	X
(ALBESHRI e CAELLI, 2010)		X			
(RANCHAL, BHARGAVA, *et al.*, 2010)		X			X
(ANGIN, BHARGAVA, *et al.*, 2010)		X	X		

Como pode ser observado, este trabalho trata da federação de identidades em um ambiente em nuvem, porém não é centrado no usuário e ainda não alcançou a identificação anônima.

Em comparação com os trabalhos relacionados, a infraestrutura obtida para prover gerenciamento de identidades e controle de acesso teve como resultados:

1. ser independente de terceiros, assim como os trabalhos de (RANCHAL, BHARGAVA, et al., 2010)(ANGIN, BHARGAVA, et al., 2010);
2. autenticar serviços em nuvem usando as políticas de privacidade do usuário, fornecendo informações mínimas para o SP;
3. garantir a proteção mútua tanto dos clientes como dos provedores. Neste trabalho destaca-se o uso de uma ferramenta específica, o Shibboleth, o qual fornece suporte às tarefas de autenticação, autorização e federação de identidades de forma prática para aplicações.

O Shibboleth se mostrou muito flexível em relação ao seu uso em um ambiente de nuvem, permitindo que um serviço seja oferecido de

forma confiável e segura. Além disso, o Shibboleth é baseado em SAML, o que significa ser compatível com padrões internacionais, garantindo, dessa forma, a interoperabilidade.

Através da configuração aplicada ao cenário, tornou-se possível o oferecimento de um serviço com permissão de acesso público, no caso de acesso somente leitura, e ao mesmo tempo com a exigência de credenciais, onde o usuário necessita estar autenticado para realizar tarefas de alteração em documentos.

5.2 LIMITAÇÔES

Em relação à implementação, este trabalho não tratou algumas questões, como por exemplo:

- o controle de acesso é realizado a nível de aplicação no provedor de serviços de forma estática;
- não é centrado no usuário;
- não foi usado XACML.

Algumas limitações do sistema Shibboleth podem ser citadas (CHADWICK, 2011):

- provê atributos de única autoridade de atributos ao provedor de serviço;
- o usuário normalmente não tem conhecimento sobre quais atributos estão sendo liberados e não consente explicitamente sobre a liberação dos atributos;
- usa “credenciais ao portador”, isto é, não contém um identificador que prove que as credenciais pertencem ao usuário.

5.3 TRABALHOS FUTUROS

Como trabalho futuro vislumbra-se um método de autorização alternativo, onde o usuário, uma vez autenticado, carrega consigo a política de acesso, e o SP deve ser capaz de interpretar estas regras. A proposta é construir um formato de tupla, a exemplo do trabalho de (CALERO, EDWARDS, *et al.*, 2010), que permita a uma empresa usar serviços da nuvem e conseguir realizar a autorização a partir do uso desta tupla. A elaboração deste formato de tupla usará alguns conceitos

importantes e pertinentes na área de controle de acesso que são: linguagem XACML, linguagem SAML, modelos de controle de acesso como o ABAC (*Attribute-Based Access Control*) e RBAC (*Role-Based Access Control*). Sendo assim, o processo de autorização deixa de ser realizado a nível de aplicação.

Outra possibilidade de pesquisa futura é expandir o cenário para representar novas formas de comunicação e, assim, criar novos casos de uso e testá-los. Por exemplo, (i) fornecer um serviço implantado em um novo SP. Considerando este serviço sendo disponibilizado por outra instituição. (ii) fornecer mais de um serviço em um mesmo SP, (iii) testar em outros ambientes de nuvem, entre outros.

Também é possível a futura investigação do uso de pseudônimos no domínio do provedor de serviço da nuvem, o que deve garantir o caráter individual dos usuários sem a necessidade de expor suas informações verdadeiras.

REFERÊNCIAS

AHN, G.-J.; LAM, J. Managing privacy preferences for federated identity management. **workshop on Digital identity management**, Virginia, 11 nov. 2005. 28-36.

AHRONOVITZ, M. et al. **Cloud Computing Use Cases White Paper Version 4.0**. [S.l.], p. 68. 2010.

ALBESHRI, A.; CAELLI, W. Mutual Protection in a Cloud Computing Environment. **High Performance Computing and Communications (HPCC)**, Melbourne, 27 set. 2010. 641-646.

AMAZON. **Amazon Elastic Compute Cloud User Guide API Version 15-12-2011**. Amazon Web Services. [S.l.]. 2011.

AMAZON. Amazon Web Services, 2012. Disponivel em: <http://aws.amazon.com/>. Acesso em: 05 mar. 2012.

ANGIN, P. et al. An Entity-Centric Approach for Privacy and Identity Management in Cloud Computing. **Reliable Distributed Systems**, New Delhi, 09 nov. 2010. 177-183.

BELAPURKAR, A. et al. **Distributed Systems Security:** Issues, Processes and Solutions. 1. ed. Chichester: John Wiley & Sons Ltd, 2009.

BERTINO, E.; TAKAHASHI, K. **Identity Management Concepts, Technologies, and Systems**. Boston: Artech House, 2011.

BHARGAV-SPANTZEL, A. et al. User centricity: A taxonomy and open issues. **Journal of Computer Security**, The Netherlands, v. 15, n. 5, p. 493-527, out. 2007. ISSN 0926-227X.

BUECKER, A. et al. **Federated Identity Management and Web Services Security with IBM Tivoli Security Solutions**. Second Edition. ed. [S.l.]: IBM Redbooks, 2005.

CALERO, J. M. A. et al. Toward a Multi-Tenancy Authorization System for Cloud Services. **Security & Privacy, IEEE**, v. 8, n. 6, p. 48-55, Dezembro 2010. ISSN 10.1109/MSP.2010.194.

CAMERON, K. THE LAWS OF IDENTITY. **Kim Cameron's Identity Weblog**, 2005. Disponivel em: <http://www.identityblog.com/?p=352/#lawsofiden_topic3>. Acesso em: 15 fev. 2012.

CARMODY, S. et al. **Technical Requirements and Information**. InCommon. [S.l.]. 2005.

CHADWICK, D. W. Federated Identity Management. In: _____ **Foundations of Security Analysis and Design V**. Heidelberg: Springer-Verlag Berlin, 2009. p. 96-120.

CHADWICK, D. W. Facilitating the Attribute Economy. **Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)**, Brasília, 2011.

CHANDRAMOULI, R.; MELL, P. State of Security Readiness. **Crossroads**, New York, v. 16, n. 3, p. 23-25, mar. 2010. ISSN 10.1145/1734160.1734168.

CHAPPELL, D. Introducing Windows CardSpace. **msdn**, 2006. Disponivel em: <http://msdn.microsoft.com/en-us/library/aa480189.aspx>. Acesso em: 17 fev. 2012.

CORDOVA, A. S. D.; WESTPHALL, C. M. **Aplicação Prática de um Sistema de Gerenciamento de Identidades**. Universidade do Vale do Itajaí. Itajaí, p. 145. 2006.

CSA. **Security Guidance for Critical Areas of Focus in Cloud Computing V2.1**. Cloud Security Alliance. [S.l.], p. 76. 2009.

CSA. **Domain 12: Guidance for Identity & Access Management V2.1**. Cloud Security Alliance. [S.l.], p. 39. 2010a.

CSA. **Guia de Segurança para Áreas Críticas Focado em Computação em Nuvem V2.1**. Cloud Security Allliance. [S.l.], p. 81. 2010b.

DAMIANI, E.; VIMERCATI, S. D. C. D.; SAMARATI, P. Managing multiple and dependable identities. **Internet Computing, IEEE**, v. 7, n. 6, p. 29-37, dez. 2003. ISSN 10.1109/MIC.2003.1250581.

DOKUWIKI. **DokuWiki**, 2012. Disponivel em: <http://www.dokuwiki.org/dokuwiki>. Acesso em: 20 fev. 2012.

GOPALAKRISHNAN, A. Cloud Computing Identity Management. **SETLabs Briefings**, 2009. 45-54.

GROBAUER, B.; WALLOSCHEK, T.; STÖCKER, E. Understanding Cloud Computing Vulnerabilities. **Security & Privacy, IEEE**, v. 9, n. 2, p. 50-57, March-April 2011. ISSN 10.1109/MSP.2010.115.

GROUP, O. Disponivel em: <http://pubs.opengroup.org/onlinepubs/9699919899/toc.pdf>. Acesso em: 12 jun. 2012.

HODGES, J.; MORGAN, R. **Lightweight Directory Access Protocol (v3): Technical Specification**. IETF. [S.l.], p. 6. 2002.

ICPP; ULD; SNG. **Identity Management Systems (IMS): Identification and Comparison Study**. Studio Notarile Genghini. Schleswig-Holstein, p. 327. 2003. (19960-2002-10).

INTERNET2. TestShib Two. **TestShib**, 2011. Disponivel em: <https://www.testshib.org/testshib-two/index.jsp>. Acesso em: 08 mar. 2012.

INTERNET2. Internet 2 Middleware. **Internet2**, 2012. Disponivel em: <http://middleware.internet2.edu/eduperson/>. Acesso em: 08 mar. 2012.

ITU. **Baseline capabilities for enhanced global identity management and interoperability**. ITU-T X.1250. [S.l.], p. 24. 2009.

ITU-T. **Information technology - Open Systems Interconnection - The Directory: Overview of concepts, models and services**. ITU-T Recommendation. [S.l.], p. 30. 2001.

JASIG. CAS. **Jasig**, 2010. Disponivel em: <http://www.jasig.org/cas>. Acesso em: 08 mar. 2012.

JøSANG, A. et al. Trust requirements in identity management. **Australian Computer Society, Inc.**, Newcastle, v. 44, p. 99-108, 2005. ISSN 1-920-68226-0.

JøSANG, A.; POPE, S. User Centric Identity Management. **AusCERT Asia Pacific Information Technology Security Conference**, Gold Coast, Maio 2005. 1-13.

JUNIPER NETWOKS. IDENTITY FEDERATION IN A HYBRID CLOUD COMPUTING ENVIRONMENT SOLUTION GUIDE, 2009. Disponivel em: <http://www.juniper.net/us/en/local/pdf/implementation-guides/8010035-en.pdf>. Acesso em: 10 Fevereiro 2012.

LEACH, P. J.; MEALLING, M.; SALZ, R. A Universally Unique IDentifier (UUID) URN Namespace. **IETF**, 2005. Disponivel em: <http://www.ietf.org/rfc/rfc4122.txt>. Acesso em: 18 fev. 2012.

LEANDRO, M. A. P. et al. Multi-Tenancy Authorization System with Federated Identity for Cloud-Based Environments Using Shibboleth. **International Conference on Networks (ICN)**, Saint Gilles, Reunion, 29 fev. 2012. 88-93.

LEE, H.; JEUN, I.; JUNG, H. Criteria for Evaluating the Privacy Protection Level of Identity Management Services. **Emerging Security Information, Systems and Technologies**, Athens, Glyfada, 21 ago. 2009. 155-160.

LIBERTY ALLIANCE. **Introduction to the Liberty Alliance Identity Architecture**. Liberty Alliance. [S.l.], p. 14. 2003.

MALER, E.; REED, D. The Venn of Identity: Options and Issues in Federated Identity Management. **Security & Privacy, IEEE**, v. 6, n. 2, p. 16-23, abr. 2008. ISSN 10.1109/MSP.2008.50.

MARCONJR, A. et al. Aspectos de segurança e privacidade em ambientes de Computação em Nuvem. **X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**, Fortaleza, 2010. 53-102.

MELL, P.; GRANCE, T. **The NIST Definition of Cloud Computing**. National Institute of Standards and Technology. [S.l.], p. 7. 2011.

MELLO, E. R. et al. A Model for Authentication Credentials Translation in Service Oriented Architecture. In: _____ **Transactions on Computational Science IV**. 10.1007/978-3-642-01004-0_5. ed. Heidelberg: Springer-Verlag Berlin, v. Transactions on Computational Science IV, 2009. p. 68-86.

MICROSOFT. Windows Azure, 2012. Disponivel em: <http://www.microsoft.com/windowsazure/>. Acesso em: 05 mar. 2012.

NASCIMENTO, T. J.; WESTPHALL, C. M. **Gerenciamento de Identidades em um Provedor de Serviços na Nuvem**. Universidade Federal de Santa Catarina. Florianópolis, p. 36. 2011.

OASIS. **Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0**. OASIS. [S.l.], p. 86. 2005.

OASIS. **eXtensible Access Control Markup Language (XACML) Version 2.0**. OASIS Standard. [S.l.], p. 141. 2005.

OASIS. **Security Assertion Markup Language (SAML) V2.0 Technical Overview**. OASIS. [S.l.], p. 51. 2008.

OASIS. **Web Services Federation Language (WSFederation) Version 1.2**. OASIS. [S.l.], p. 140. 2009.

OPEN GROUP. **Risk Taxonomy**. The Open Group. Berkshire, United Kingdom , p. 39. 2009.

OPENID. OpenId. **OpenId**, 2010. Disponivel em: <http://openid.net>. Acesso em: 17 fev. 2012.

OPENLDAP FOUNDATION. OpenLDAP, 2012. Disponivel em: <http://www.openldap.org/>. Acesso em: 08 mar. 2012.

ORACLE. MySQL. **SQL**, 2012. Disponivel em: <http://www.mysql.com/>. Acesso em: 08 mar. 2012.

PALSONKENNEDY, R.; GOPAL, T. V. Assessing the risks and opportunities of Cloud Computing — Defining identity management systems and maturity models. **Trendz in Information Sciences & Computing (TISC)**, Chennai, 17 fev. 2011. 138-142.

RACKSPACE. The Rackspace Cloud. **Rackspace**, 2012. Disponivel em: <http://www.rackspace.com/cloud/>. Acesso em: 06 mar. 2012.

RANCHAL, R. et al. Protection of Identity Information in Cloud Computing without Trusted Third Party. **Reliable Distributed Systems**, New Delhi, 09 nov. 2010. 368-372.

RNP. brEduPerson. **Wiki RNP**, 2010. Disponivel em: <http://wiki.rnp.br/display/cafewebsite/brEduPerson>. Acesso em: 08 mar. 2012.

RNP. Procedimentos Operacionais da Federação CAFe. **Wiki RNP**, 2011. Disponivel em: <http://wiki.rnp.br/pages/viewpage.action?pageId=41190088>. Acesso em: 08 mar. 2012.

SANTOS, A. L. D. **Quem Mexeu no meu Sistema?** 1. ed. Rio de Janeiro: Brasport, 2008.

SANTOS, D. R. D.; WESTPHALL, C. M. **Gerenciamento de Identidades e Privacidade em ambientes de Computação em Nuvem**. Universidade Federal de Santa Catarina. Florianópolis, p. 83. 2011.

SCAVO, T.; CANTOR, S. Shibboleth Architecture, p. 31, 2005. Disponivel em: <http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-tech-overview-latest.pdf>. Acesso em: 18 fev. 2012.

SHIBBOLETH. Shibboleth 2.x - Confluence. **Wiki Shibboleth**, 2011. Disponivel em: <https://wiki.shibboleth.net/confluence/display/SHIB2/Home>. Acesso em: 08 mar. 2012.

SOUSA, F. R. C.; MOREIRA, L. O.; MACHADO, J. C. Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios. **ERCEMAPI**, Parnaíba, out. 2009.

SPRING. Open Cloud Manifesto, 2009. Disponivel em: <http://www.opencloudmanifesto.org/opencloudmanifesto1.htm>. Acesso em: 12 fev. 2012.

SWITCH. uApprove. **Serving Swiss Universities**, 2012. Disponivel em: <http://www.switch.ch/aai/support/tools/uApprove.html>. Acesso em: 08 mar. 2012.

ZHOU, M. et al. Services in the Cloud Computing Era: A Survey. **Universal Communication Symposium (IUCS), 2010 4th International**, Beijing, 13 dez. 2010. 40-46.